CASS REPORT # 93-3-01

# IMAGING RADAR STUDIES OF ATMOSPHERIC WINDS AND WAVES

by

Kent L. Miller, R.G. Roper, G.W. Adams

and John W. Brosnahan

August 30, 1993



FINAL REPORT: AIRFORCE OFFICE OF SCIENTIFIC RESEARCH
CONTRACT # F49620-92-J-0193



UTAH STATE UNIVERSITY

CENTER FOR ATMOSPHERIC AND SPACE SCIENCES
UTAH STATE UNIVERSITY • LOGAN • UTAH • 84322

## CASS REPORT # 93-3-01

# IMAGING RADAR STUDIES OF ATMOSPHERIC WINDS AND WAVES

by

Kent L. Miller, R.G. Roper, G.W. Adams

and John W. Brosnahan

August 30, 1993

FINAL REPORT: AIRFORCE OFFICE OF SCIENTIFIC RESEARCH
CONTRACT # F49620-92-J-0193



UTAH STATE UNIVERSITY

CENTER FOR ATMOSPHERIC AND SPACE SCIENCES
UTAH STATE UNIVERSITY • LOGAN • UTAH • 84322

# REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188



1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
Sept. 2, 1993

3. REPORT TYPE AND DATES COVERED
Final Report-1 Mar ~~92~~ 2 - 31 Aug 93

4. TITLE AND SUBTITLE
Imaging Radar Studies of Atmospheric Winds and Waves

5. FUNDING NUMBERS
F49620-92-J-0193
61102F
2310
CS

6. AUTHOR(S)
Kent L. Miller, R. G. Roper, G. W. Adams and John W. Brosnahan

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Center for Atmospheric and Space Sciences
Utah State University
Logan, UT 84322-4405

8. PERFORMING ORGANIZATION REPORT NUMBER
CASS Report #93-3-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NL
110 Duncan Avenue Suite B115
Bolling AFB, DC 20332-0001
Maj Kroll

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unrestricted

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

In this report, further analyses of data taken during the Arecibo Initiative in Dynamics of the Atmosphere (AIDA'89) campaigns is presented, concentrating on the third campaign (Scene III) from May 2 through May 9, 1989. The major emphasis is on the comparison between the MAPSTAR Imaging Doppler Interferometry (IDI) radar and the Arecibo Observatory incoherent scatter (ISR) radar, but some comparisons with the Geospace Corporation meteor wind radar (MWR), and the Arecibo Observatory Fabry-Perot spectrometer (FPS) are also included. A reanalysis of four IDI - ISR - MWR comparisons show better agreement between all three techniques than previously reported. In general, better agreement was found between the daytime radar winds than that previously published for the April campaign, with the zonal winds agreeing better than the meridional. In contrast, the nighttime IDI - FPS comparisons showed better agreement in the meridional component, and considerably better agreement after midnight than before.

14. SUBJECT TERMS

15. NUMBER OF PAGES
298

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT

# IMAGING RADAR STUDIES OF ATMOSPHERIC WINDS AND WAVES

## PREFACE

The untimely death of Dr. Gene Adams, the original Principal Investigator, occurred on March 14, 1992, just two weeks after the initiation of this contract. This report is dedicated to his memory.

## ABSTRACT

In this report, we present further analyses of data taken during the Arecibo Initiative in Dynamics of the Atmosphere (AIDA'89) campaigns, concentrating on the third campaign (Scene III) from May 2 through May 9, 1989.The major emphasis is on the comparison between the MAPSTAR imaging Doppler interferometry (IDI) radar and the Arecibo Observatory incoherent scatter (ISR) radar, but some comparisons with the Geospace Corporation meteor wind radar (MWR), and the Arecibo Observatory Fabry-Perot spectrometer (FPS) are also included. A reanalysis of four IDI - ISR - MWR comparisons shows better agreement between all three techniques than previously reported. In general, we find better agreement between the daytime radar winds than that previously published for the April campaign, with the zonal winds agreeing better than the meridional. In contrast, the nighttime IDI - FPS comparisons show better agreement in the meridional component, and considerably better agreement after midnight than before.

[The report consists of two parts - Part 1, in which plots are presented and discussed, and Part 2, which details the data reduction and analysis procedures, and contains listings of the computer programs used].

## STUDENTS GRADUATED UNDER THIS CONTRACT

Captain R. Scott Turek, an AFIT student in the Center for Atmospheric and Space Science, successfully defended his doctoral thesis entitled "Radar Studies of the Middle Atmosphere: Morphology and Comparison of Techniques" on June 28, 1993, and is now a professor at the Air Force Academy, Colorado Springs, Colorado.

## PUBLICATIONS THIS CONTRACT PERIOD

Brosnahan, J.W., and G.W. Adams, "The MAPSTAR Imaging Doppler Interferometry (IDI) radar: description and first results," J. Atmos. Terrest Phys., 55, 203 - 228, 1993

Roper, R.G., G.W. Adams and J.W. Brosnahan, "Tidal winds at mesopause altitudes over Arecibo (18°N, 67°W), April 5 - 11, 1989 (AIDA'89)", J. Atmos. Terrest. Phys., 289-312, 1993

# Contents

## Part 1 - Results from AIDA Scene III - May 2 - 9, 1993

### IDI - ISR Comparisons

Since the submission of the papers published in the AIDA Special Issue of the Journal of Atmospheric and Terrestrial Physics in March, 1993, some of the analysis criteria for both the Arecibo Observatory incoherent scatter (ISR) and MAPSTAR imaging doppler interferometry (IDI) radars have been changed.This is illustrated in the results of the IDI - ISR - meteor wind radar (MWR) comparisons presented in Figure 1a-d, the same intervals as plotted in Figure 5 of Hines et al. (1993). We have here included results from not only the 393° azimuth soundings, but also those at 303° azimuth (note that the velocities at these azimuths are the line of sight at 11.3° zenith angle), and the zonal and meridional components inferred from these measurements. Note that, unlike the Hines et al. plots, the IDI results have been produced by projecting the IDI three dimensional wind vector onto the ISR lines of sight and then using the resultant projections into the horizontal to calculate the zonal and meridional wind velocities. This parrallels the ISR reduction, which considers the vertical wind to be zero when producing the horizontal projections. Also plotted are the velocity components inferred from a Groves analysis of the IDI data, fitting mean, diurnal and semidiurnal components to the IDI scattering point parameters for each 24 hours (from noon to noon) of the campaign. These are plotted as IDIG.

The next section concentrates on the reduction and analysis of the May 2 - 9, 1989 IDI - ISR comparisons. Figures 2 through 33 are similar plots to those presented in Figure 1 (but note that no MWR data is available for these intervals). Instead of the two hour intervals published in Hines et al. (1993) and Roper et al. (1993), the intervals used here span some 43 minutes, with the ISR radar integrating data for some 25 minutes at the 393° (123°) azimuth, and for 11 minutes at the 303° (213°) azimuth, with six to seven minutes spent in moving the gondola to change the viewing azimuth. The IDI data has been averaged for the total 43 minutes - the 11 minute timeslot is marginal for IDI wind profile determination, although the 25 minute line of sight measurements have been subject to closer scrutiny and will be published in Turek et al. (1993).

We are in the process of interpreting these results. Of interest is the apparently better agreement between the IDI and ISR velocities in the zonal (and 303° and 123° azimuths - those closer to the east - west direction) than the meridional (and 393° and 213° azimuths). This warrants further investigation.

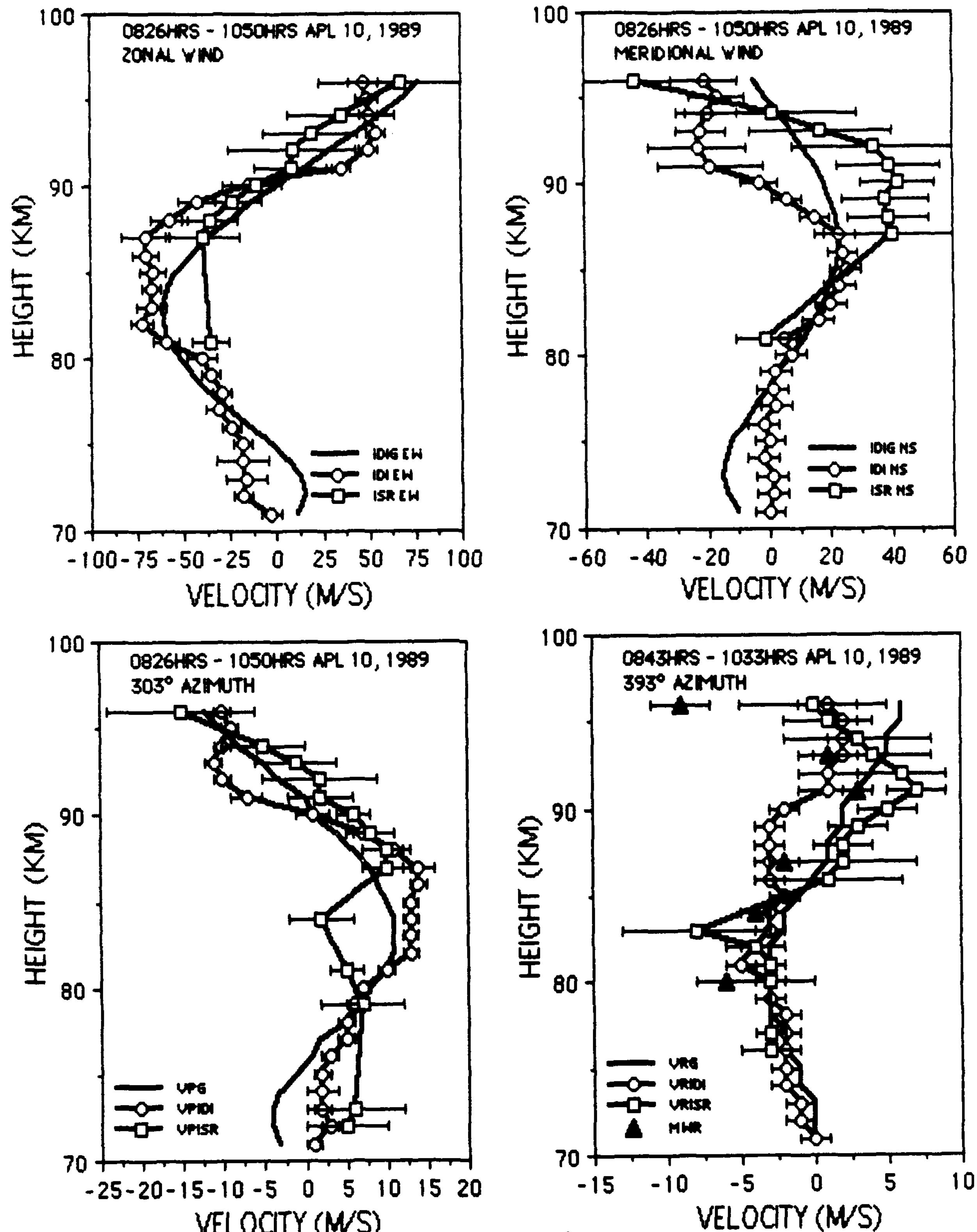

0826HRS - 1050HRS APL 10, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
IDIG EW
IDI EW
ISR EW
0826HRS - 1050HRS APL 10, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
IDIG NS
IDI NS
ISR NS
0826HRS - 1050HRS APL 10, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VPG
VPIDI
VPISR
0843HRS - 1033HRS APL 10, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VRG
VRIDI
VRISR
MWR

FIGURE 1-a

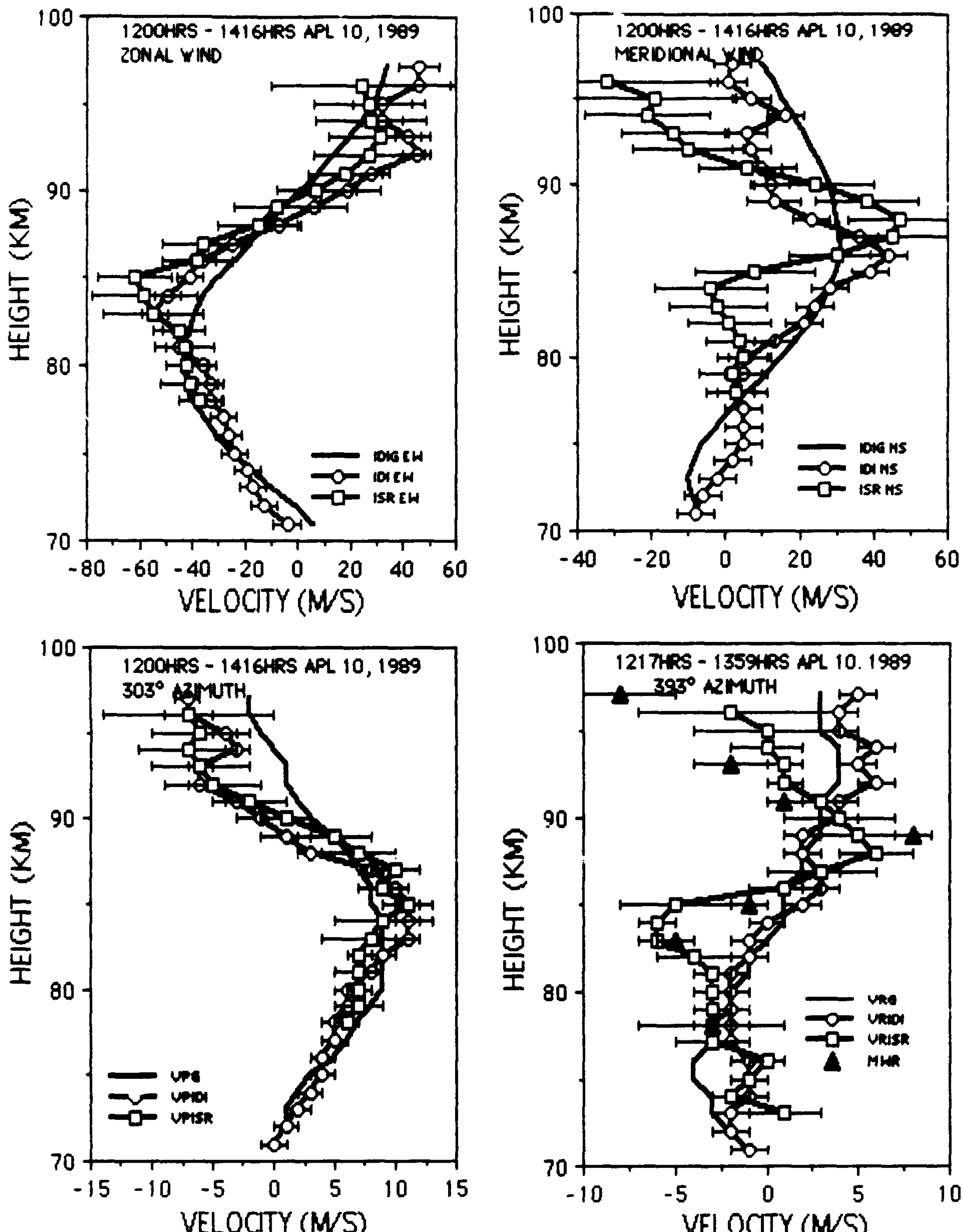

1200HRS - 1416HRS APL 10, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1200HRS - 1416HRS APL 10, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1200HRS - 1416HRS APL 10, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1217HRS - 1359HRS APL 10. 1989
393° AZIMUTH
VRG
VRIDI
VRISR
MWR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 1-b

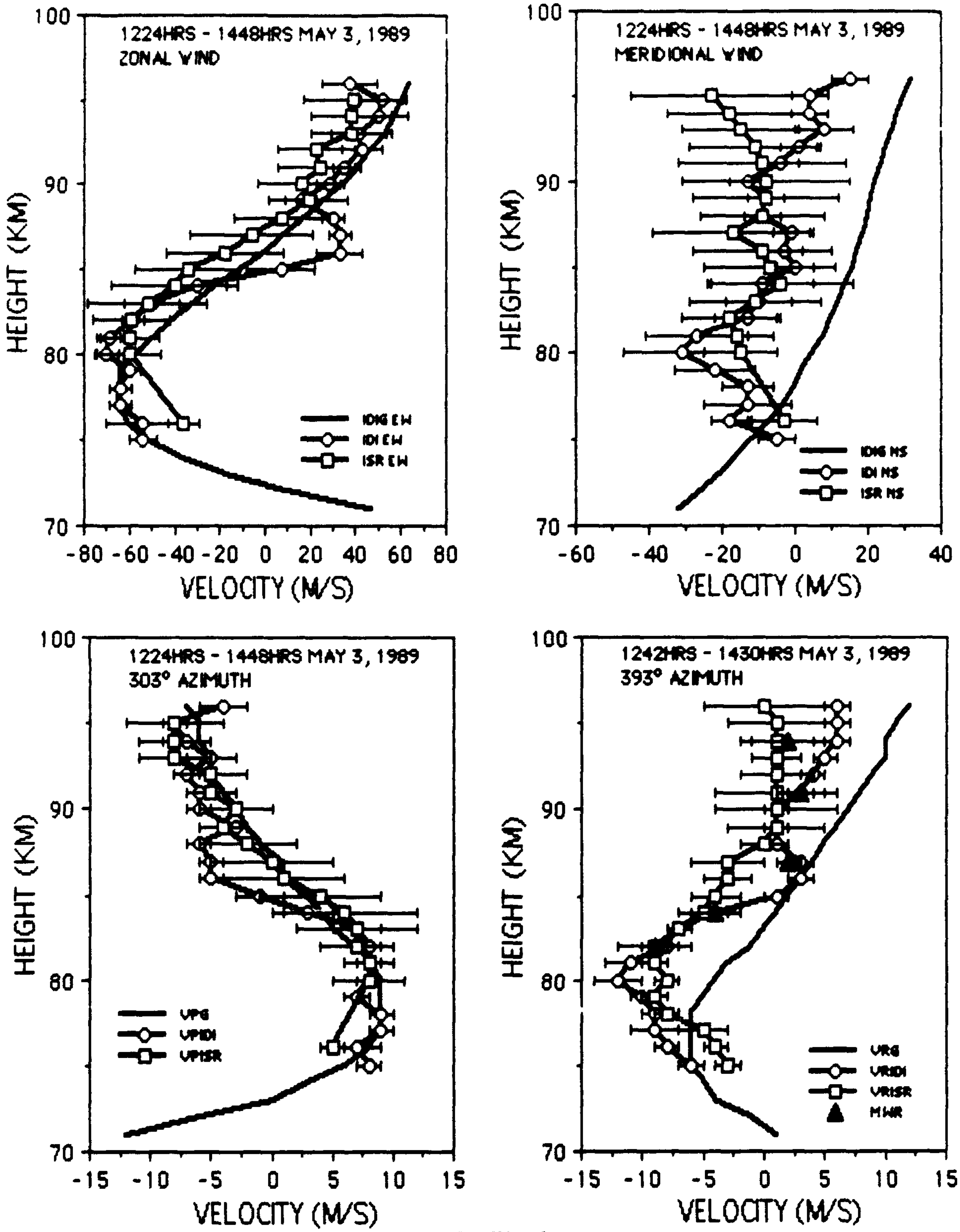

1224HRS - 1448HRS MAY 3, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1224HRS - 1448HRS MAY 3, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1224HRS - 1448HRS MAY 3, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1242HRS - 1430HRS MAY 3, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
MWR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 1-c

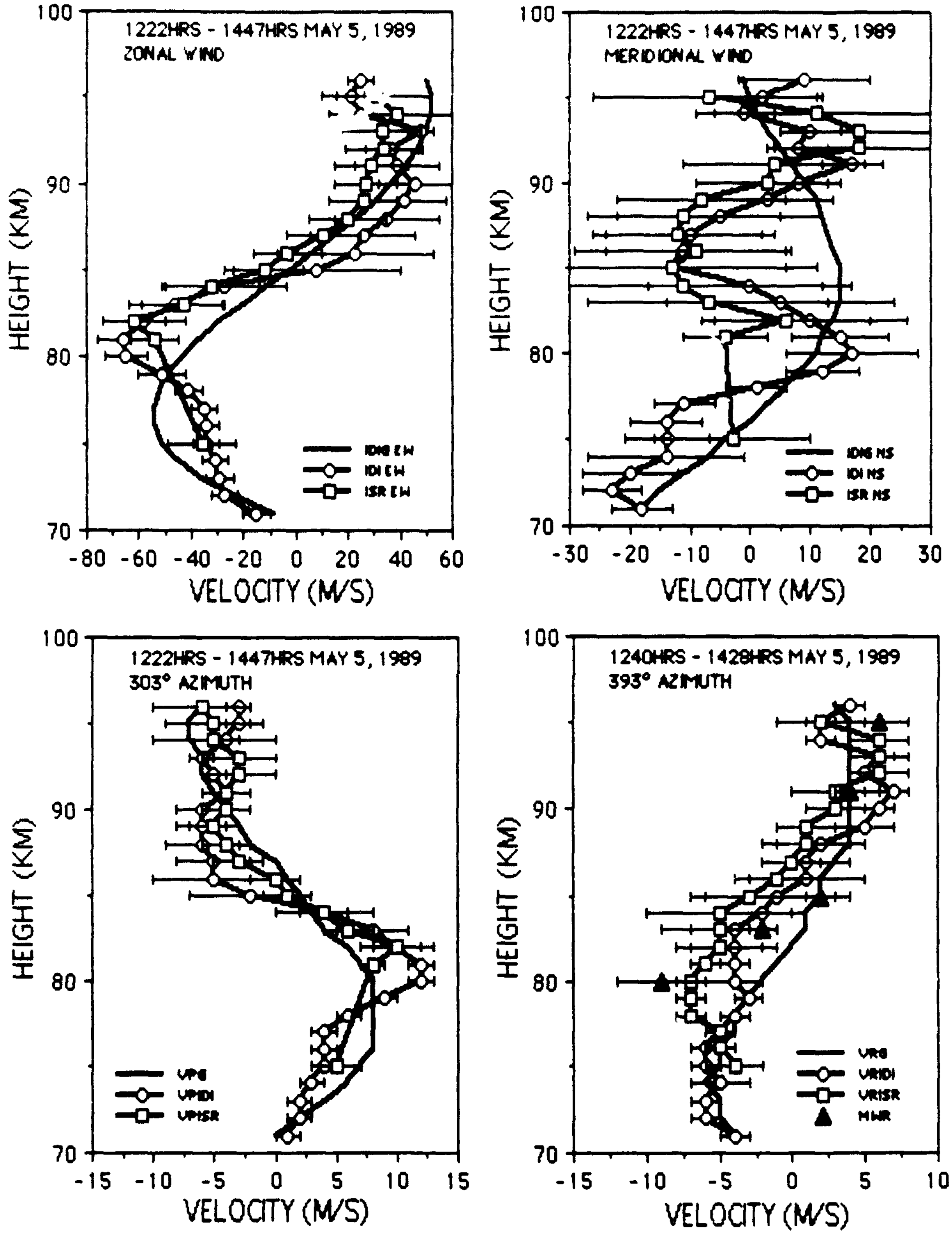

1222HRS - 1447HRS MAY 5, 1989
ZONAL WIND
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1222HRS - 1447HRS MAY 5, 1989
MERIDIONAL WIND
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1222HRS - 1447HRS MAY 5, 1989
303° AZIMUTH
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1240HRS - 1428HRS MAY 5, 1989
393° AZIMUTH
VRIDI
VRISR
MWR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 1-d

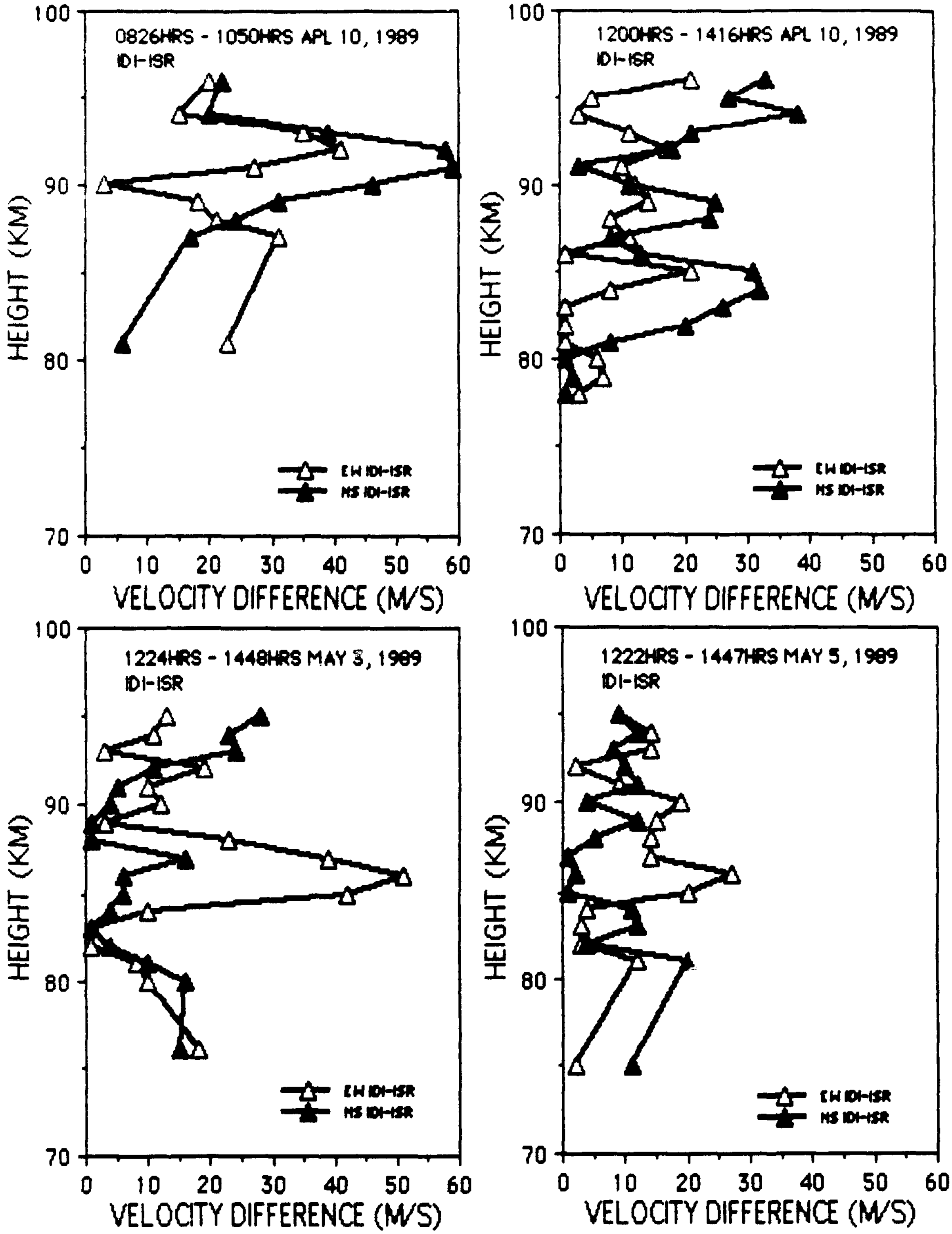

0826HRS - 1050HRS APL 10, 1989
IDI-ISR
1200HRS - 1416HRS APL 10, 1989
IDI-ISR
1224HRS - 1448HRS MAY 3, 1989
IDI-ISR
1222HRS - 1447HRS MAY 5, 1989
IDI-ISR
EW IDI-ISR
NS IDI-ISR
HEIGHT (KM)
VELOCITY DIFFERENCE (M/S)
100
90
80
70
0
10
20
30
40
50
60

FIGURE 1-e

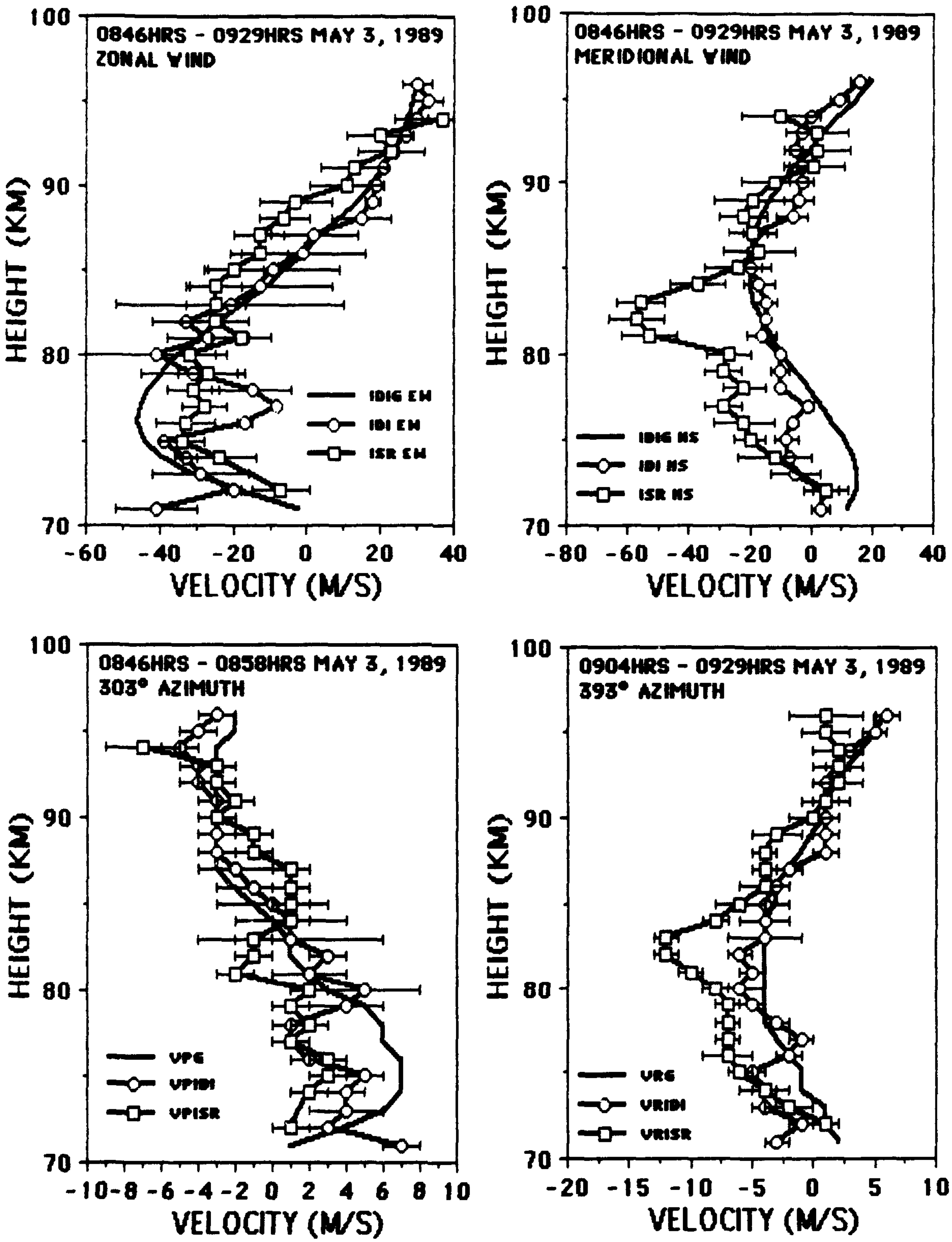

0846HRS - 0929HRS MAY 3, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
0846HRS - 0929HRS MAY 3, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
0846HRS - 0858HRS MAY 3, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
0904HRS - 0929HRS MAY 3, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 2

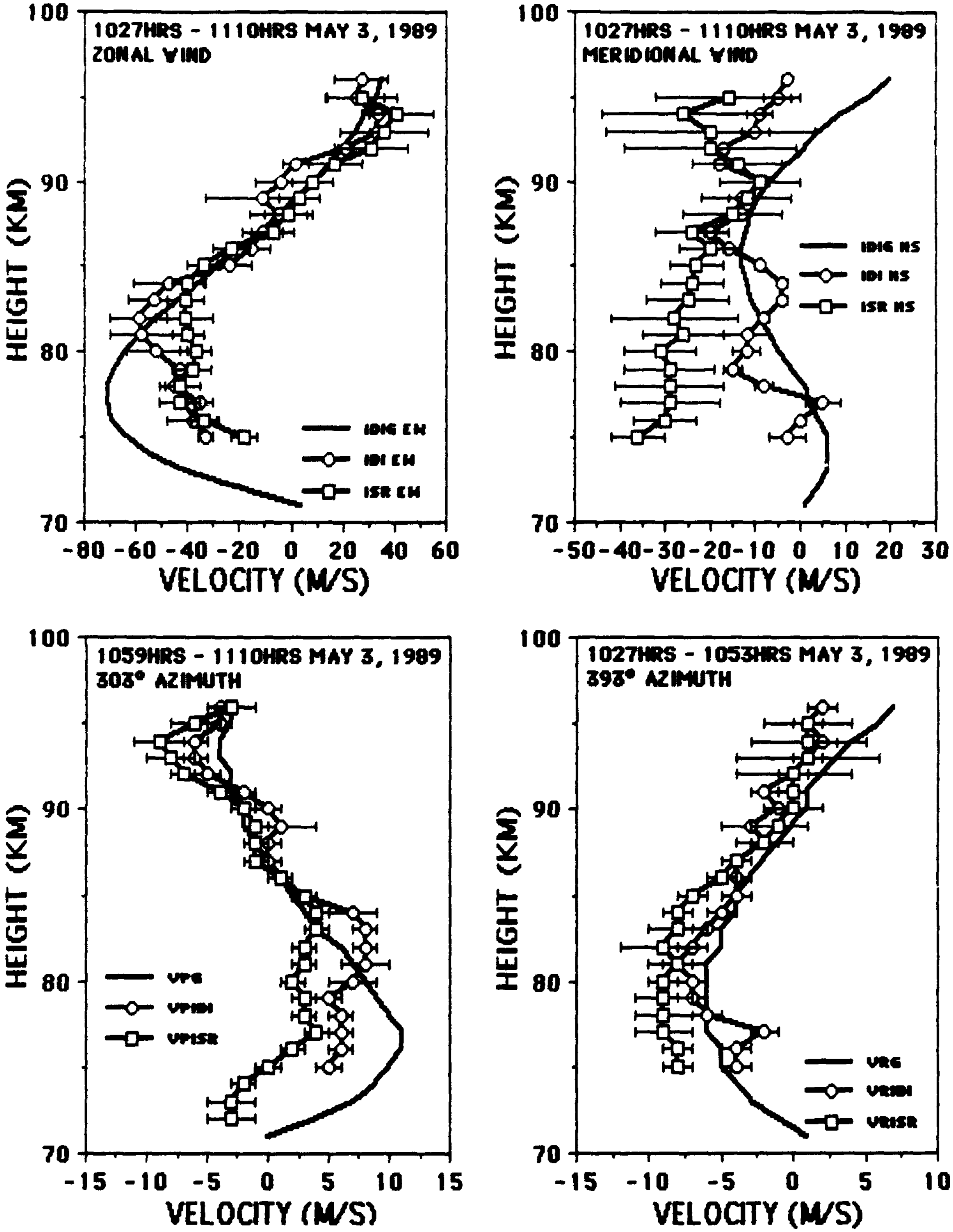

1027HRS - 1110HRS MAY 3, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1027HRS - 1110HRS MAY 3, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1059HRS - 1110HRS MAY 3, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1027HRS - 1053HRS MAY 3, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 3

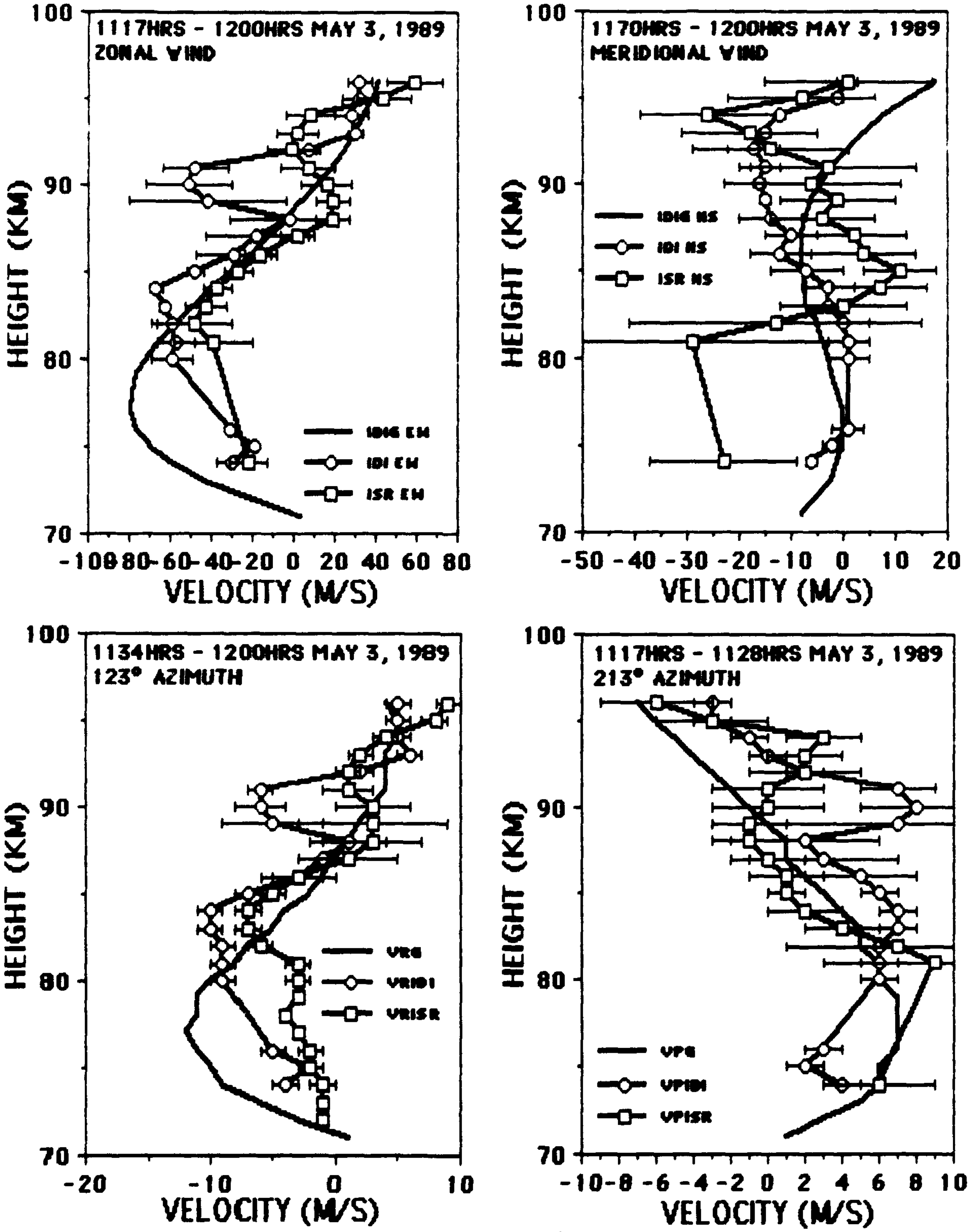

1117HRS - 1200HRS MAY 3, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1170HRS - 1200HRS MAY 3, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
1134HRS - 1200HRS MAY 3, 1989
123° AZIMUTH
VRG
VRIDI
VRISR
1117HRS - 1128HRS MAY 3, 1989
213° AZIMUTH
VPG
VPIDI
VPISR

FIGURE 4

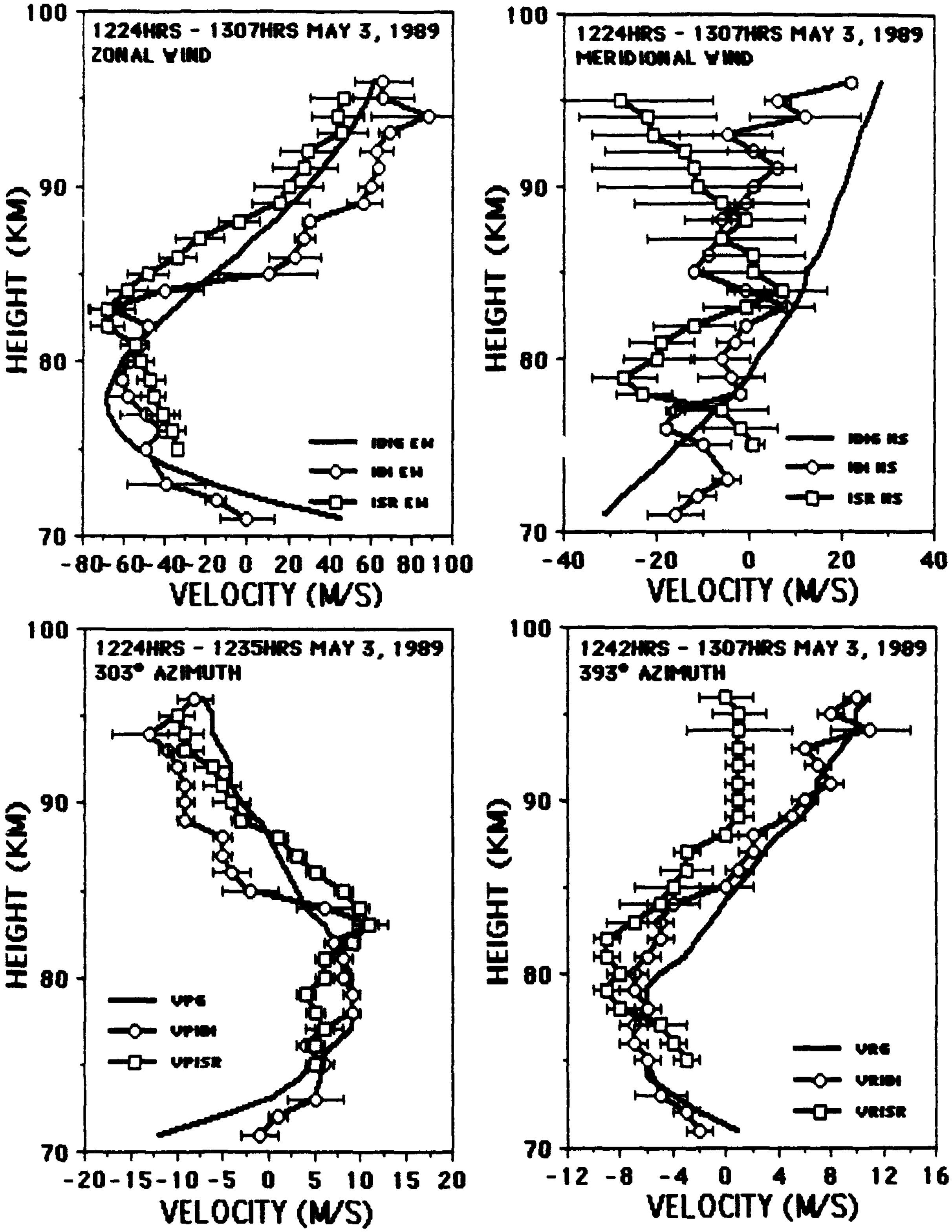
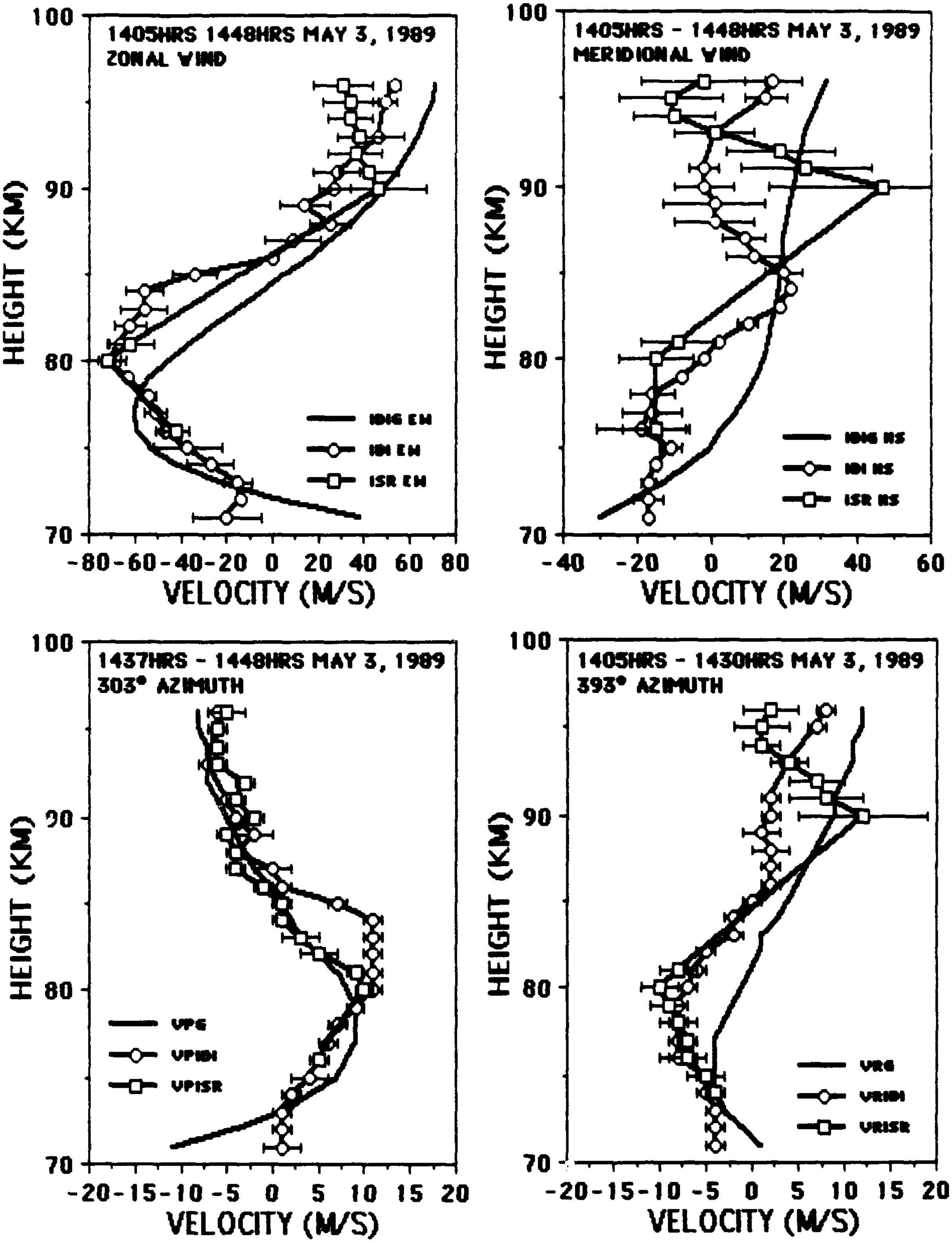

1224HRS - 1307HRS MAY 3, 1989
ZONAL WIND
1224HRS - 1307HRS MAY 3, 1989
MERIDIONAL WIND
1224HRS - 1235HRS MAY 3, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
1242HRS - 1307HRS MAY 3, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 5



1405HRS 1448HRS MAY 3, 1989
ZONAL WIND
1405HRS - 1448HRS MAY 3, 1989
MERIDIONAL WIND
1437HRS - 1448HRS MAY 3, 1989
303° AZIMUTH
1405HRS - 1430HRS MAY 3, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)


FIGURE 6

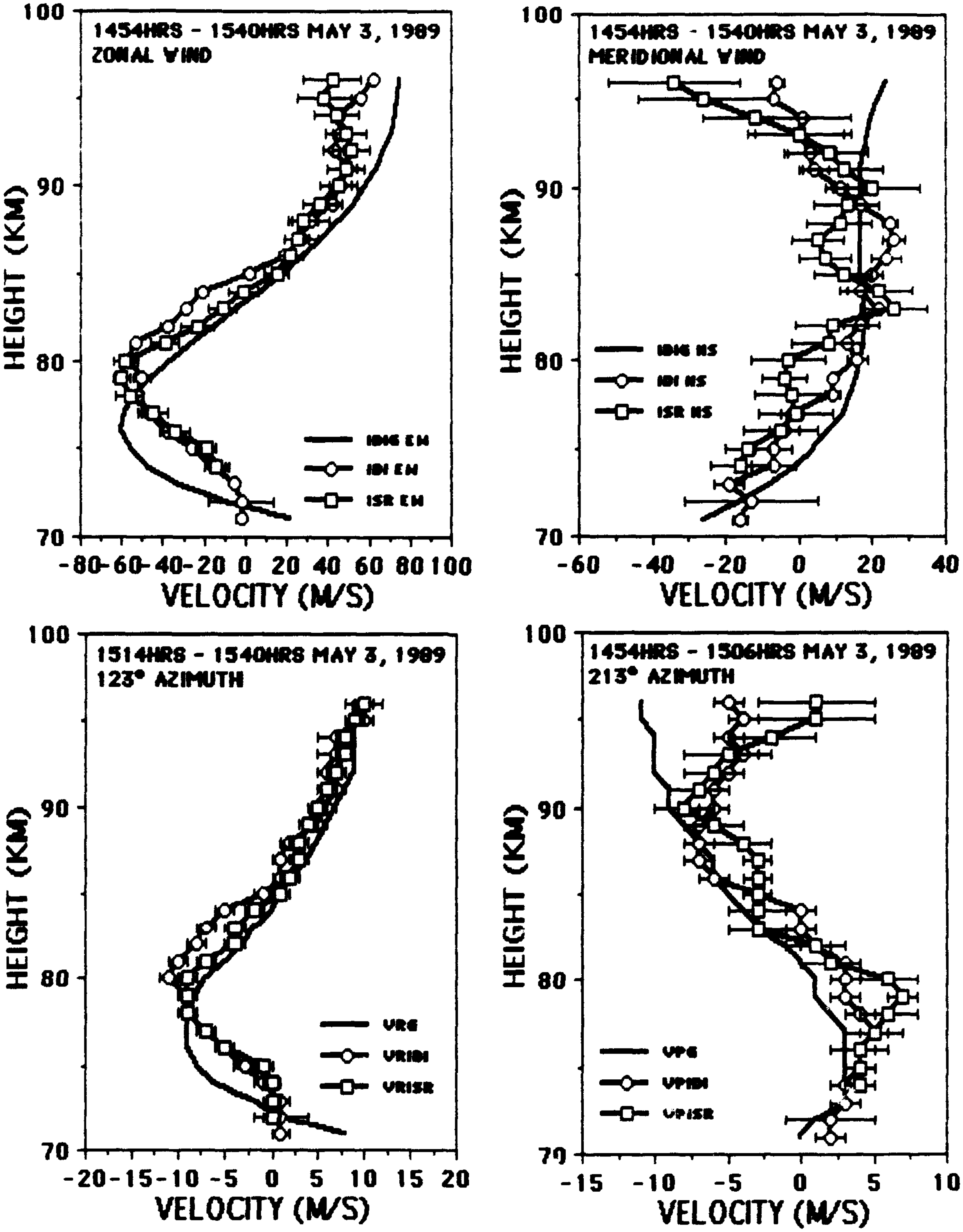

1454HRS - 1540HRS MAY 3, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1454HRS - 1540HRS MAY 3, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1514HRS - 1540HRS MAY 3, 1989
123° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VRG
VRIDI
VRISR
1454HRS - 1506HRS MAY 3, 1989
213° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VPG
VPIDI

FIGURE 7

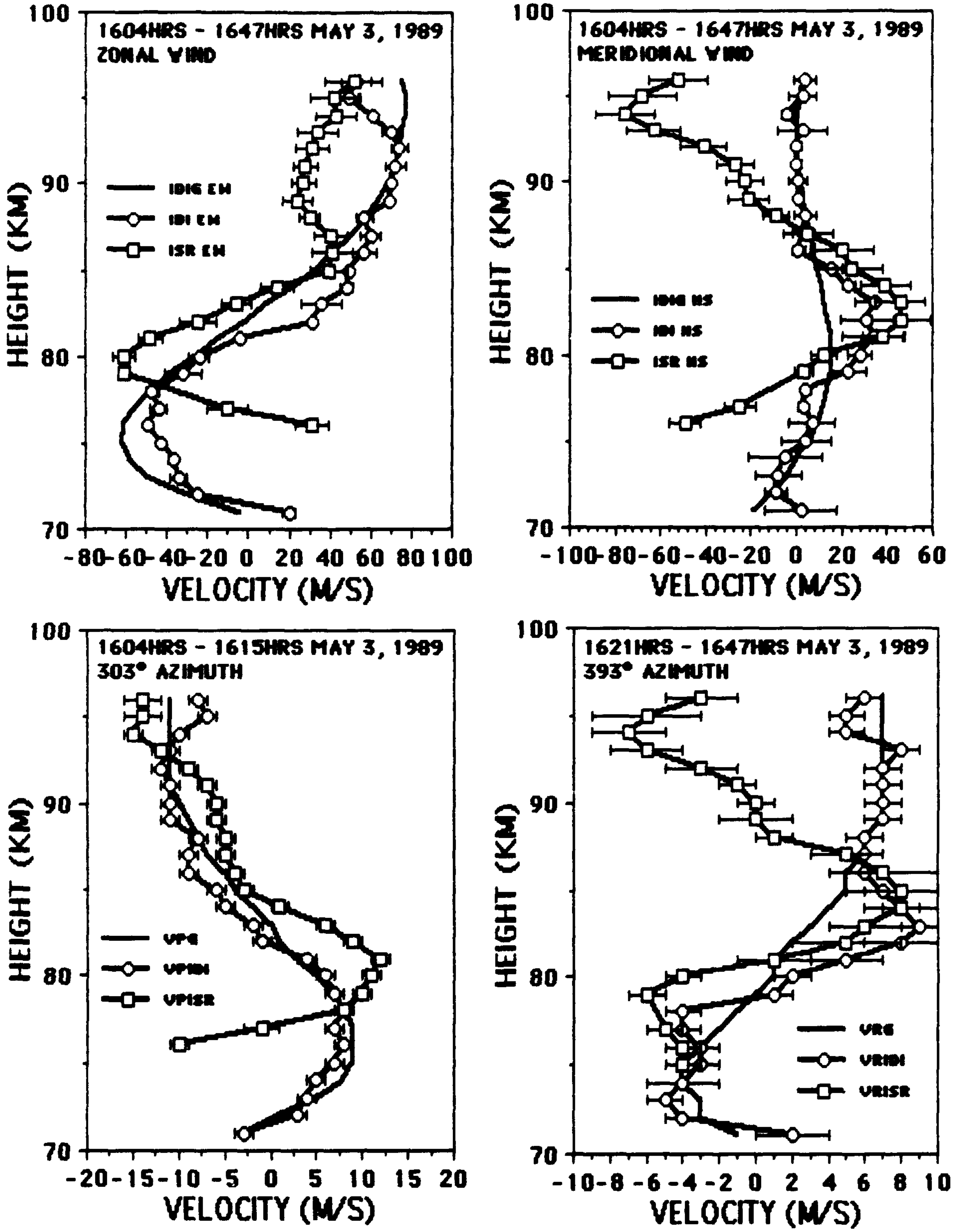

1604HRS - 1647HRS MAY 3, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1604HRS - 1647HRS MAY 3, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1604HRS - 1615HRS MAY 3, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
1621HRS - 1647HRS MAY 3, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 8

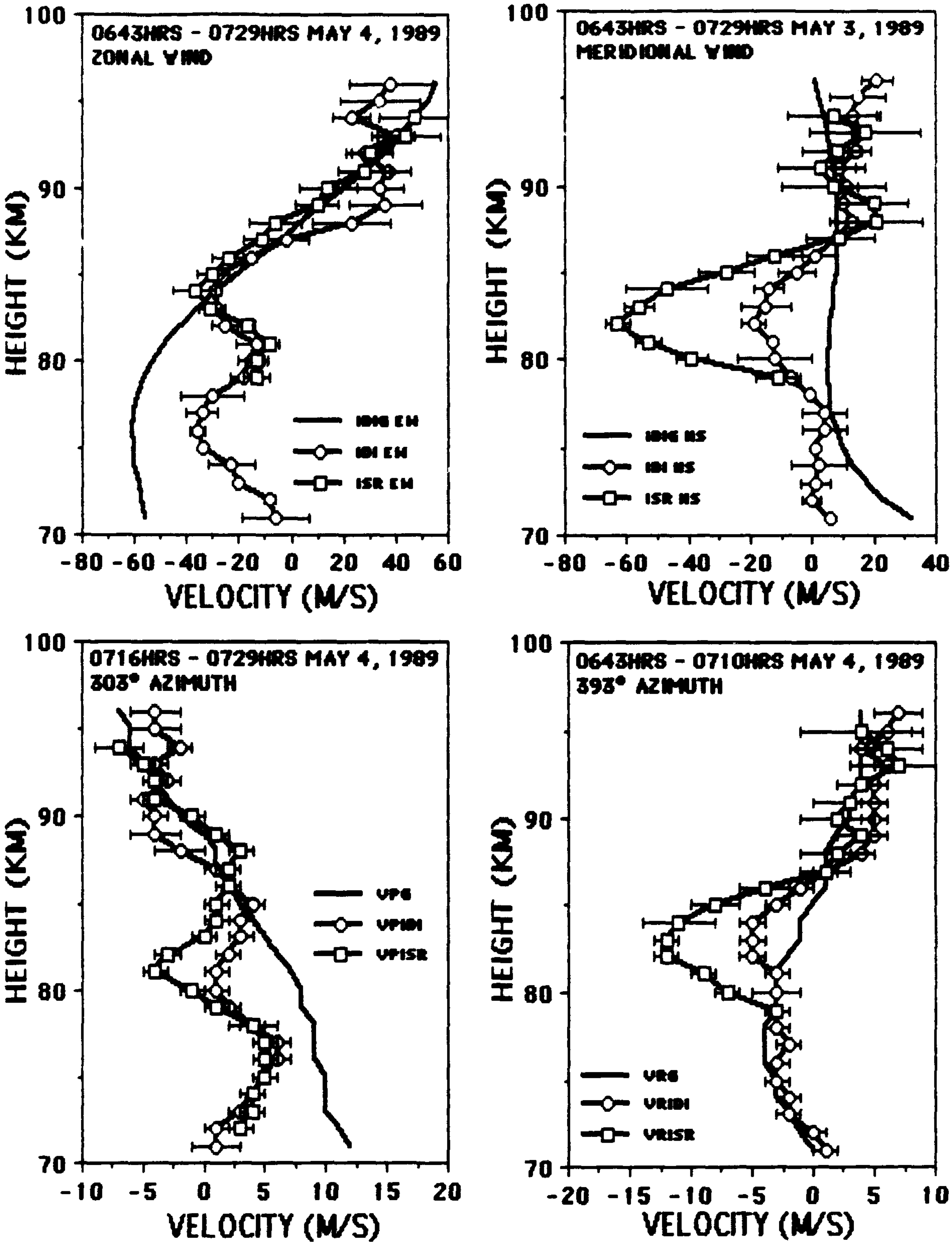

0643HRS - 0729HRS MAY 4, 1989
ZONAL WIND
IDI EW
ISR EW
0643HRS - 0729HRS MAY 3, 1989
MERIDIONAL WIND
IDI NS
ISR NS
0716HRS - 0729HRS MAY 4, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
0643HRS - 0710HRS MAY 4, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 9

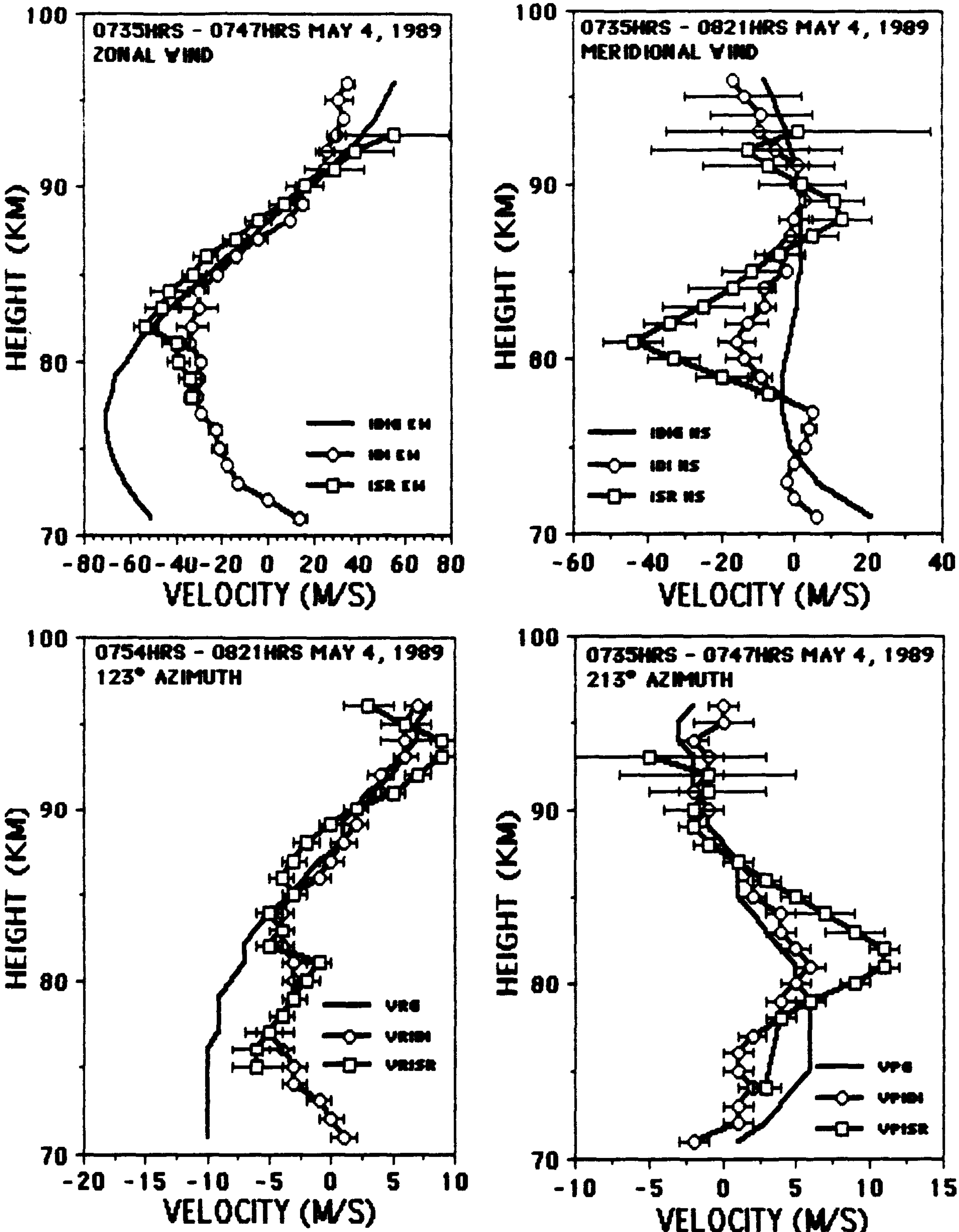
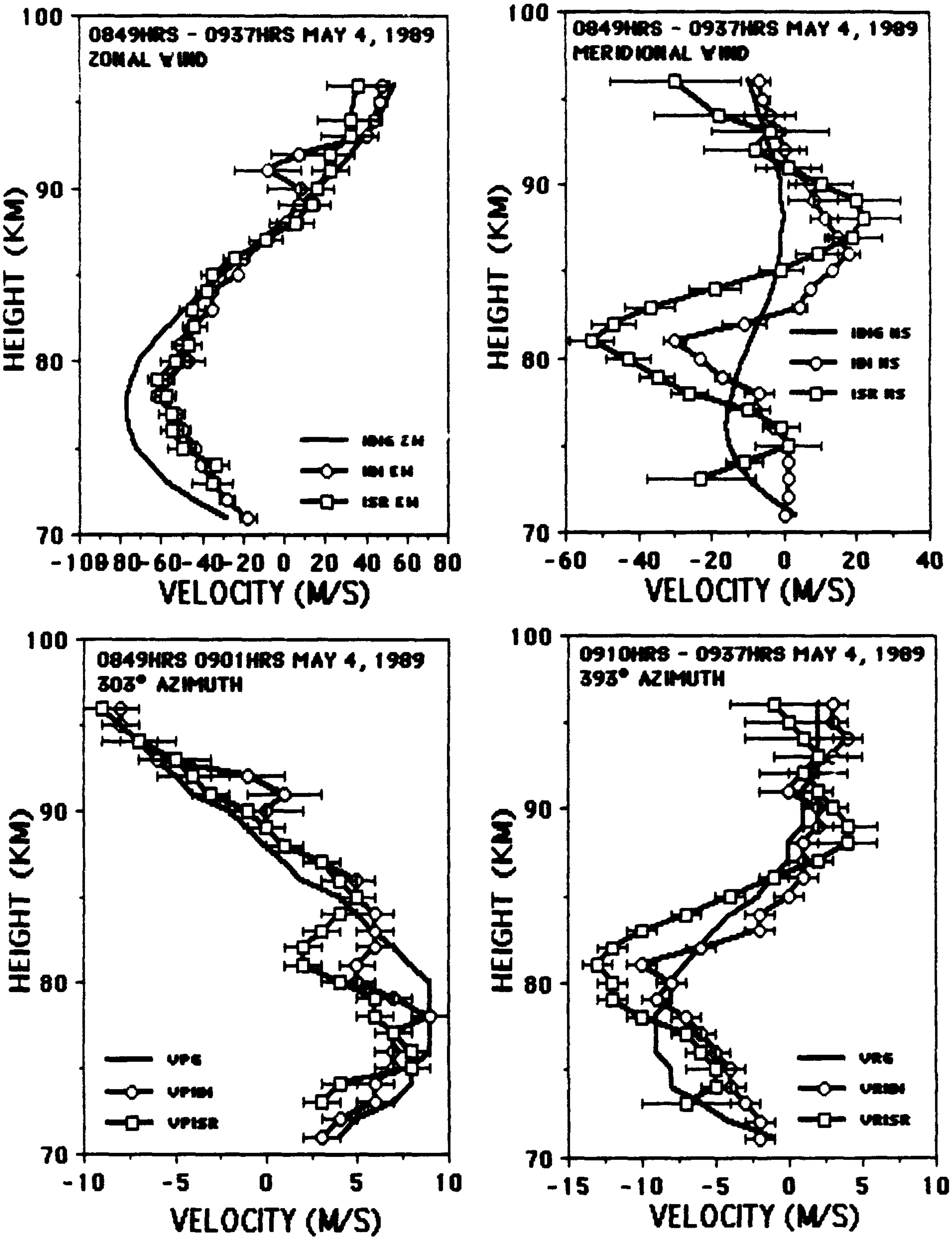

0735HRS - 0747HRS MAY 4, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0735HRS - 0821HRS MAY 4, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0754HRS - 0821HRS MAY 4, 1989
123° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
0735HRS - 0747HRS MAY 4, 1989
213° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 10



0849HRS - 0937HRS MAY 4, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0849HRS - 0937HRS MAY 4, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0849HRS 0901HRS MAY 4, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
0910HRS - 0937HRS MAY 4, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)


FIGURE 11

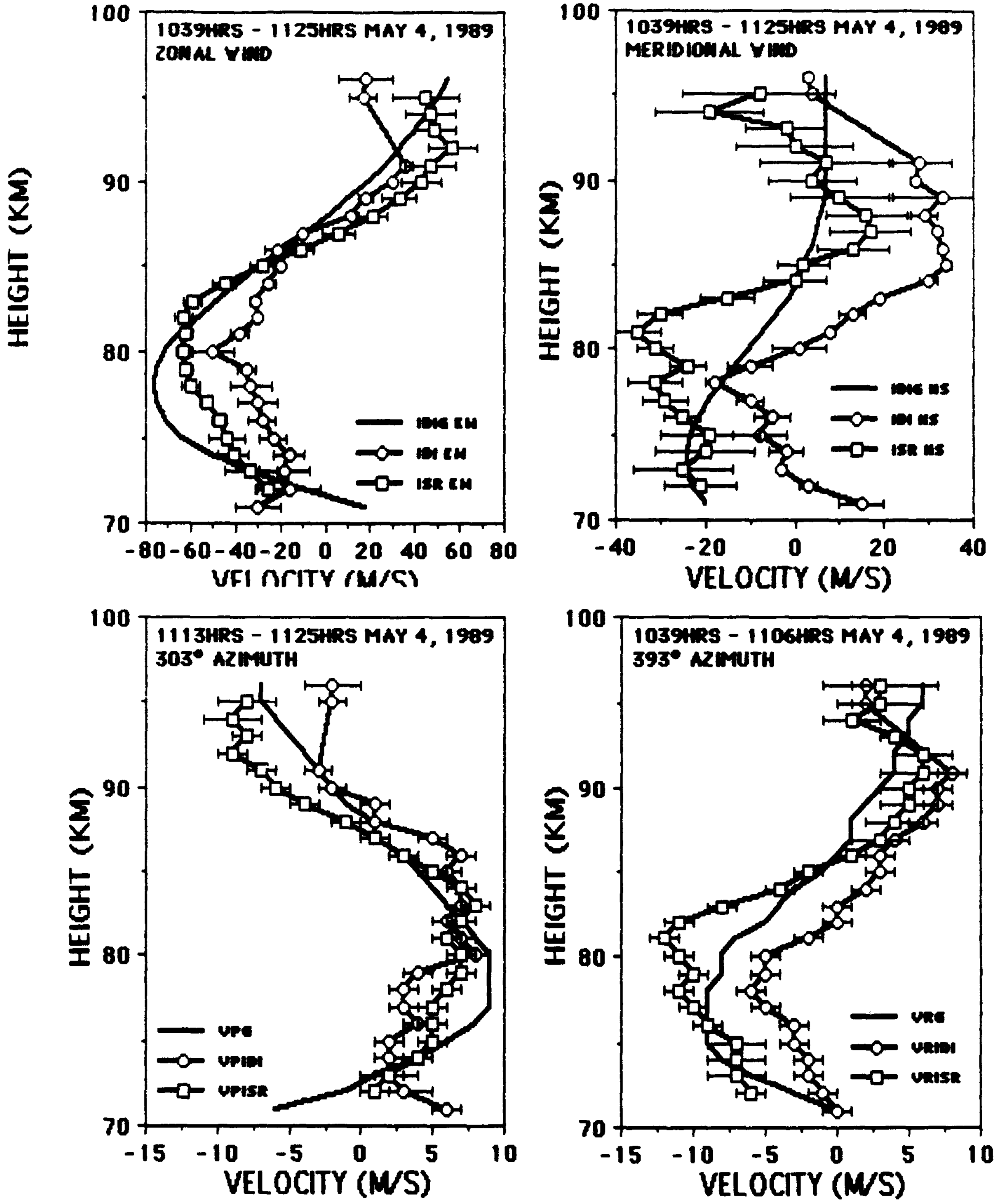
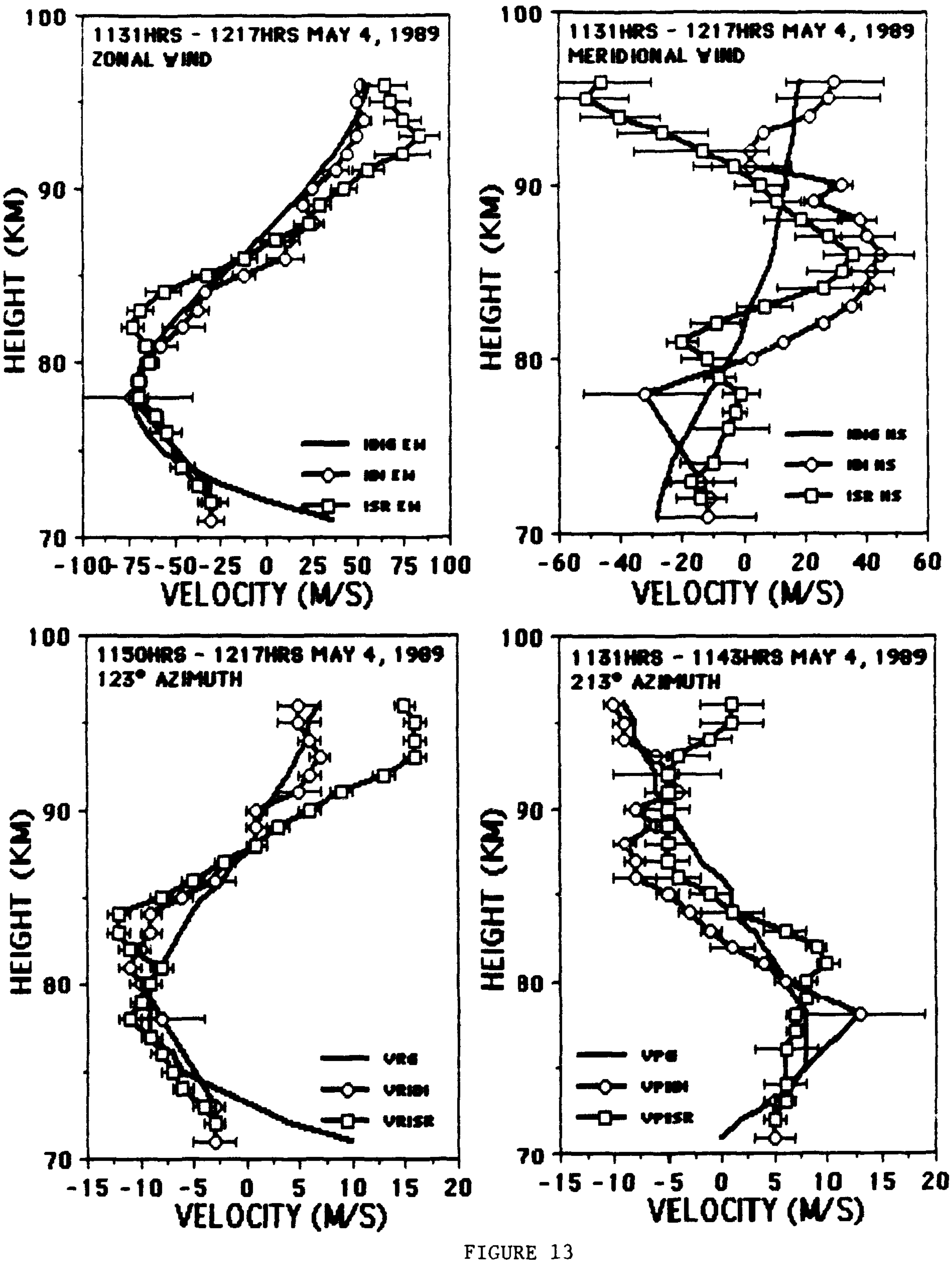

1039HRS - 1125HRS MAY 4, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1039HRS - 1125HRS MAY 4, 1989
MERIDIONAL WIND
1113HRS - 1125HRS MAY 4, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
1039HRS - 1106HRS MAY 4, 1989
393° AZIMUTH
VRG
VRIDI
VRISR

FIGURE 12



1131HRS - 1217HRS MAY 4, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1131HRS - 1217HRS MAY 4, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1150HRS - 1217HRS MAY 4, 1989
123° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
1131HRS - 1143HRS MAY 4, 1989
213° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)


FIGURE 13

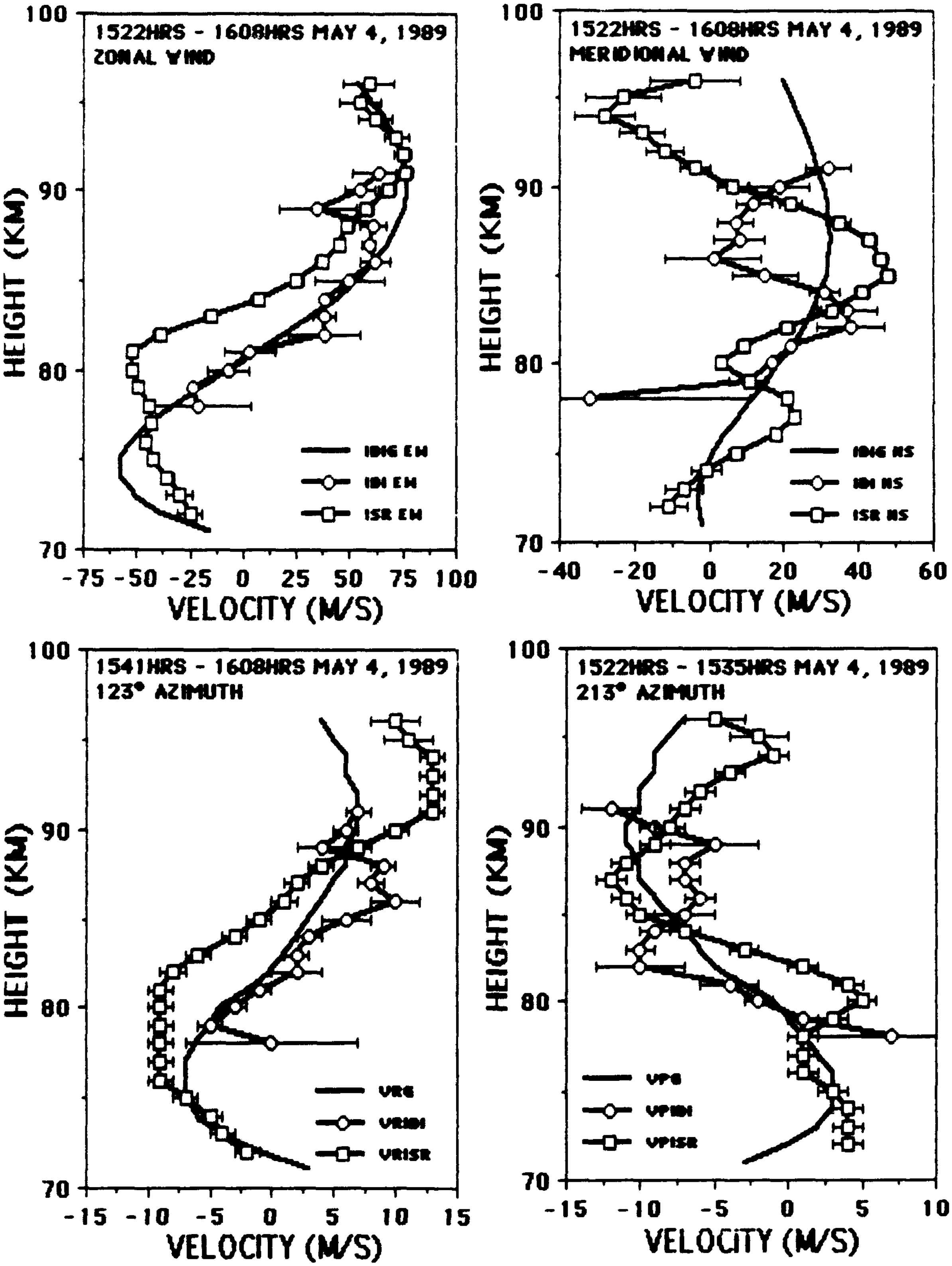

1522HRS - 1608HRS MAY 4, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1522HRS - 1608HRS MAY 4, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1541HRS - 1608HRS MAY 4, 1989
123° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
1522HRS - 1535HRS MAY 4, 1989
213° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 14

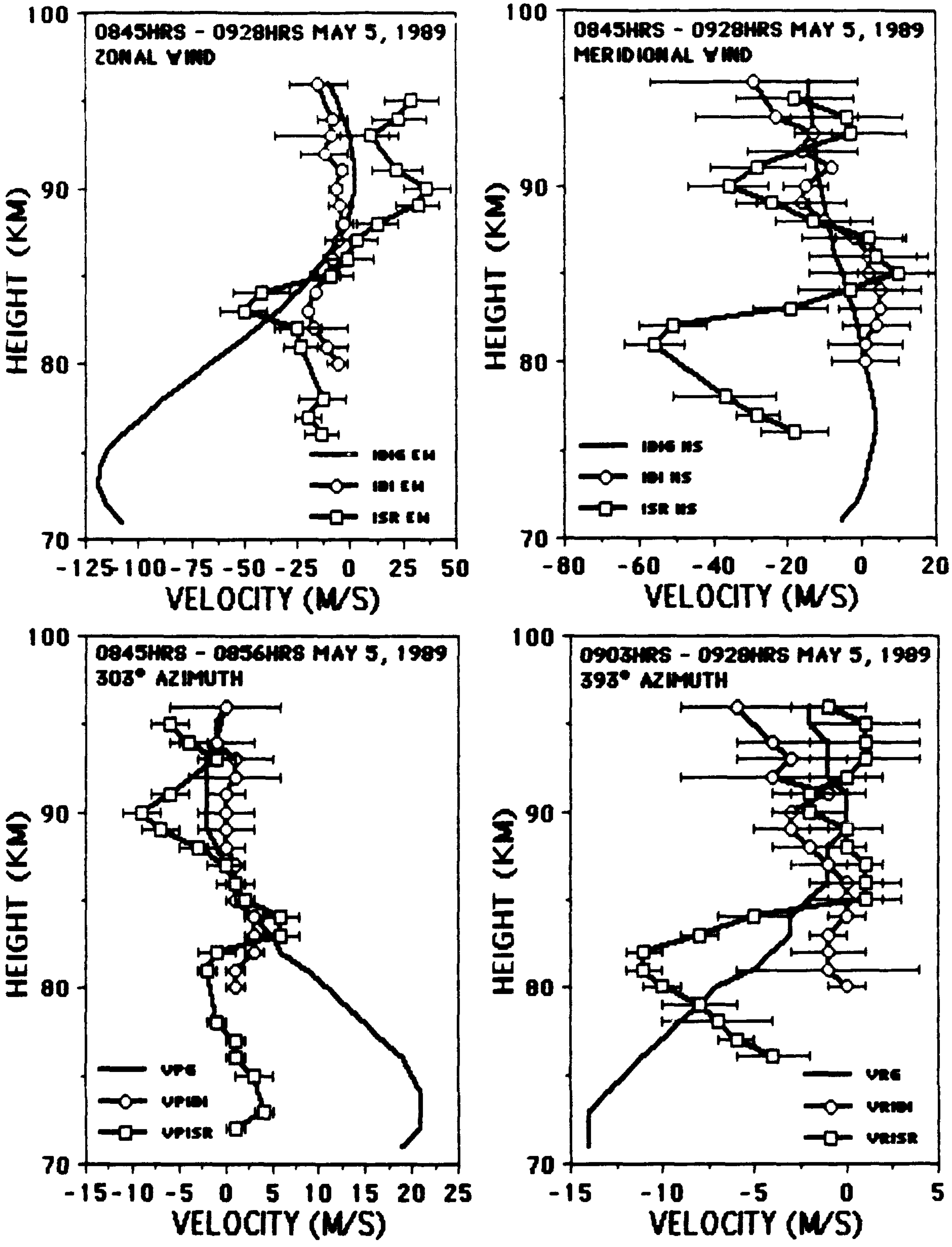

0845HRS - 0928HRS MAY 5, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
IDIG EW
IDI EW
ISR EW
-125 -100 -75 -50 -25 0 25 50
0845HRS - 0928HRS MAY 5, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
IDIG NS
IDI NS
ISR NS
-80 -60 -40 -20 0 20
0845HRS - 0856HRS MAY 5, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VPG
VPIDI
VPISR
-15 -10 -5 0 5 10 15 20 25
0903HRS - 0928HRS MAY 5, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VRG
VRIDI
VRISR
-15 -10 -5 0 5
100
90
80
70

FIGURE 15

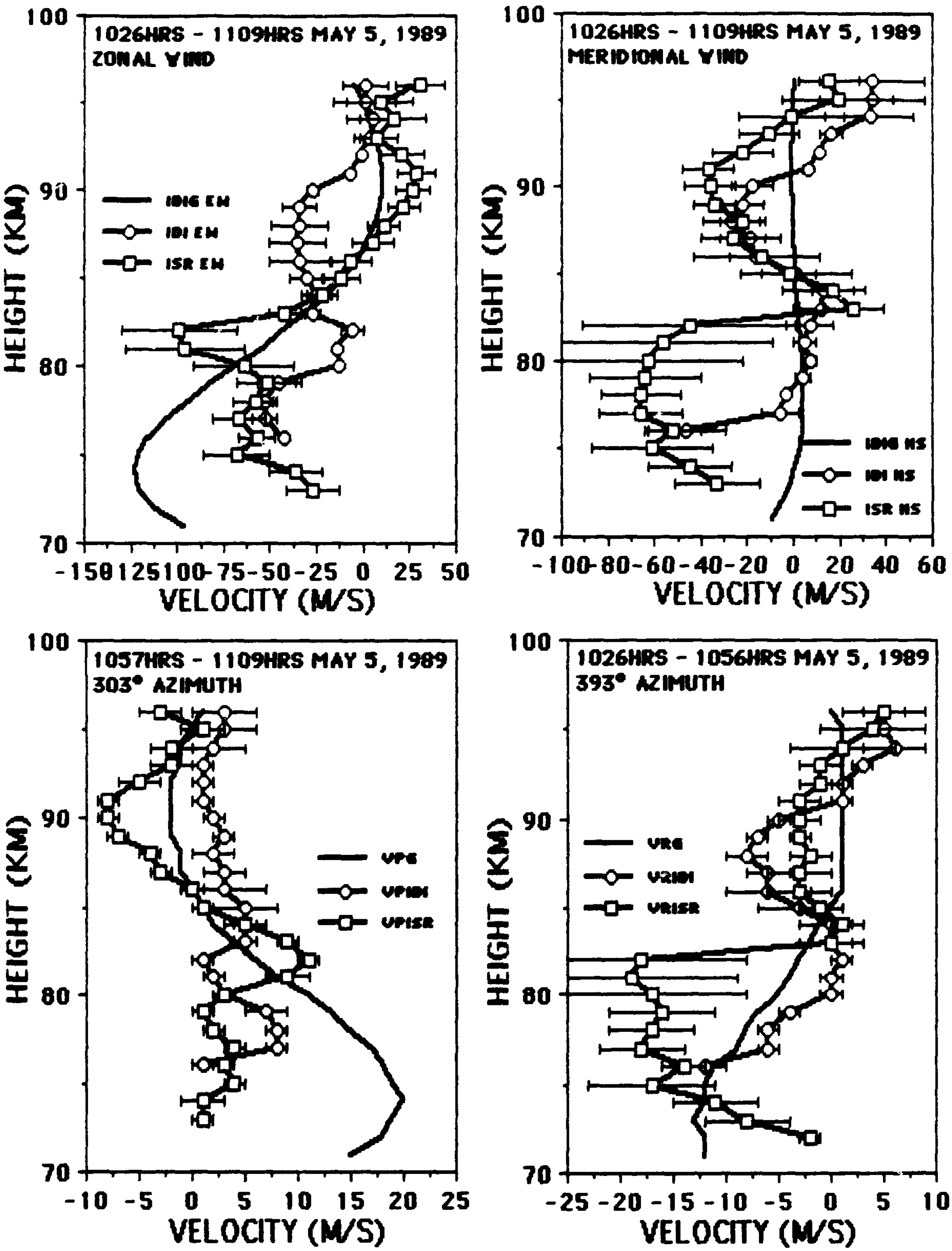

1026HRS - 1109HRS MAY 5, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1026HRS - 1109HRS MAY 5, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1057HRS - 1109HRS MAY 5, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
1026HRS - 1056HRS MAY 5, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 16

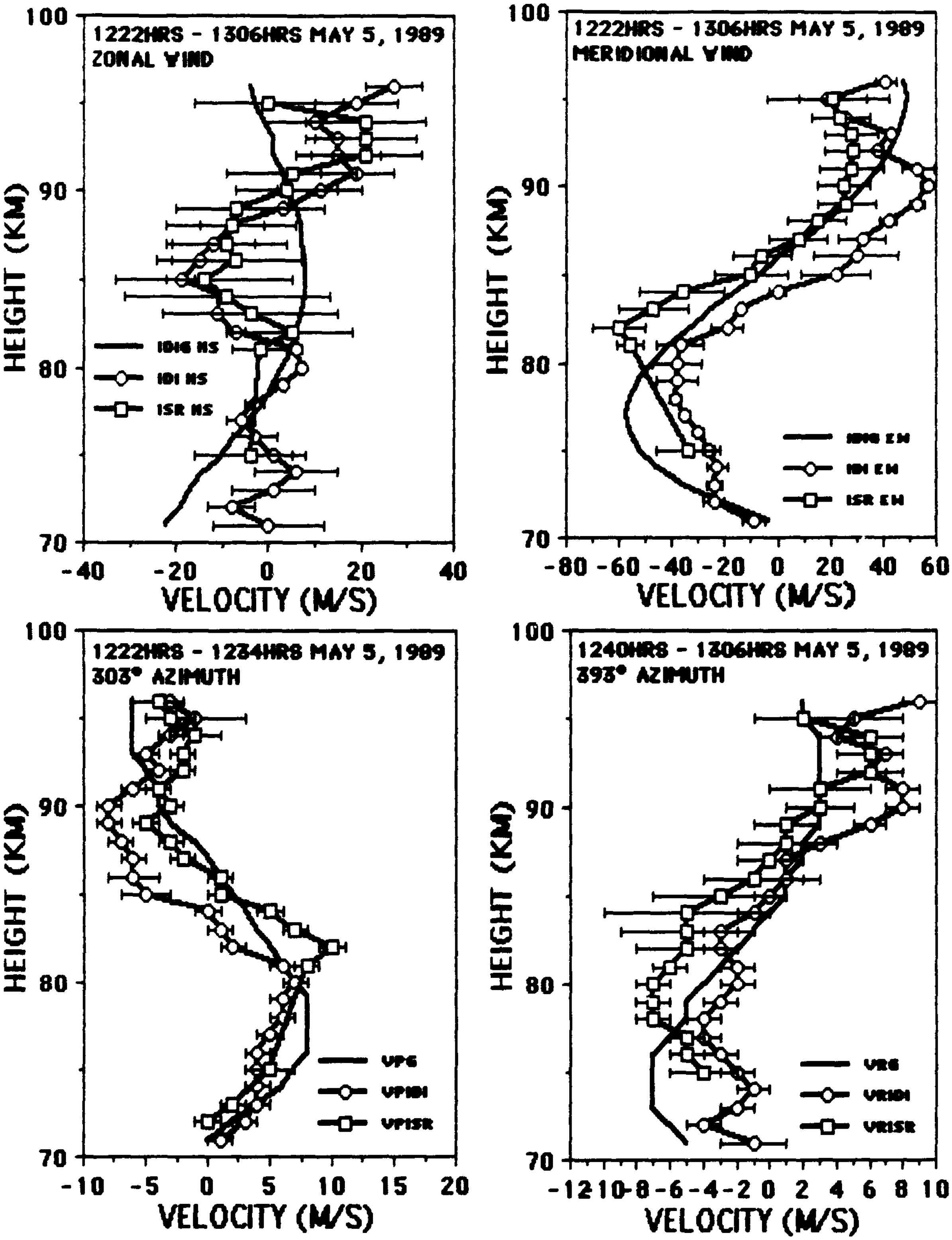
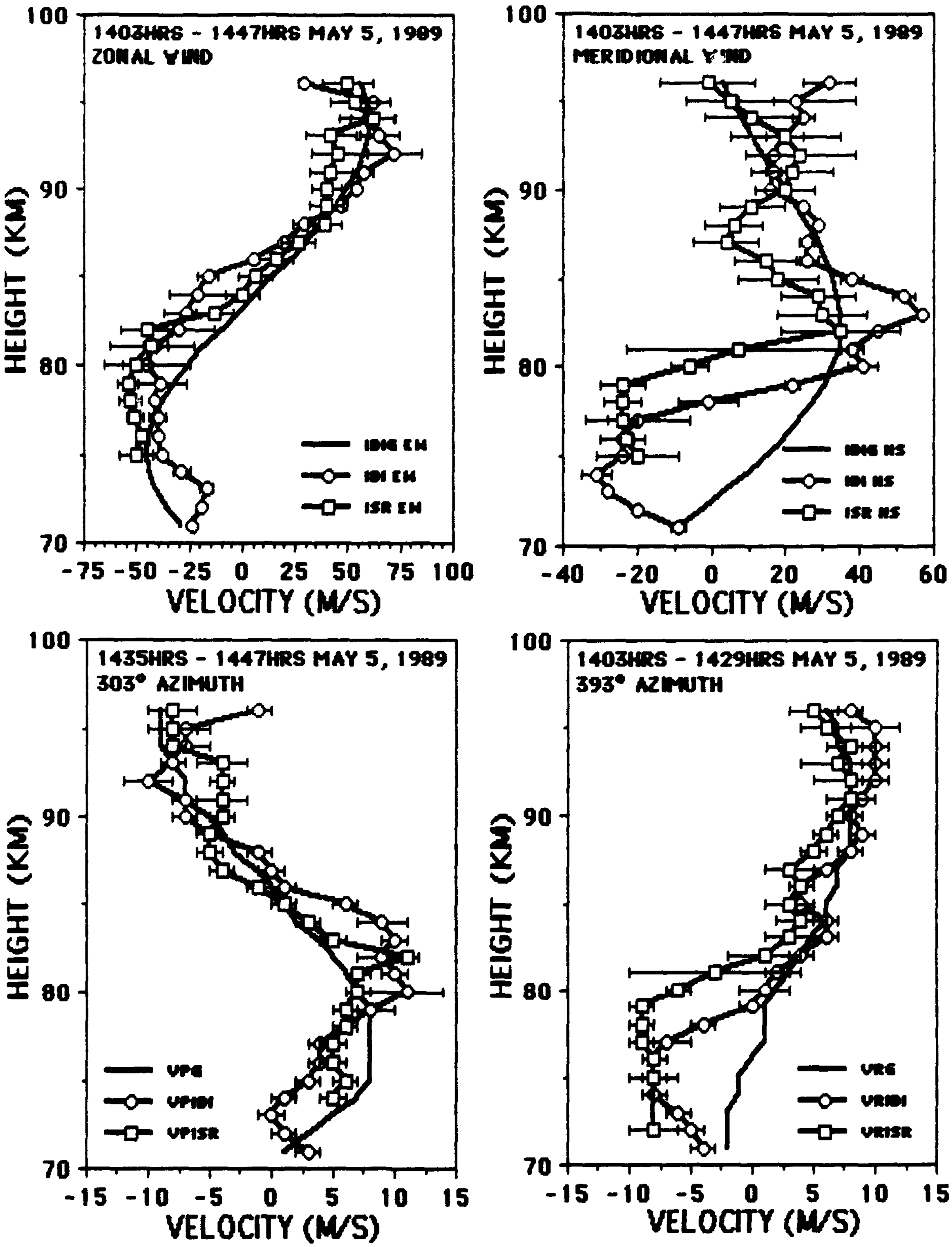

1222HRS - 1306HRS MAY 5, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1222HRS - 1306HRS MAY 5, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1222HRS - 1234HRS MAY 5, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
1240HRS - 1306HRS MAY 5, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 17



1403HRS - 1447HRS MAY 5, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
100
90
80
70
-75 -50 -25 0 25 50 75 100
1403HRS - 1447HRS MAY 5, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
-40 -20 0 20 40 60
1435HRS - 1447HRS MAY 5, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
-15 -10 -5 0 5 10 15
1403HRS - 1429HRS MAY 5, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)
-15 -10 -5 0 5 10 15


FIGURE 18

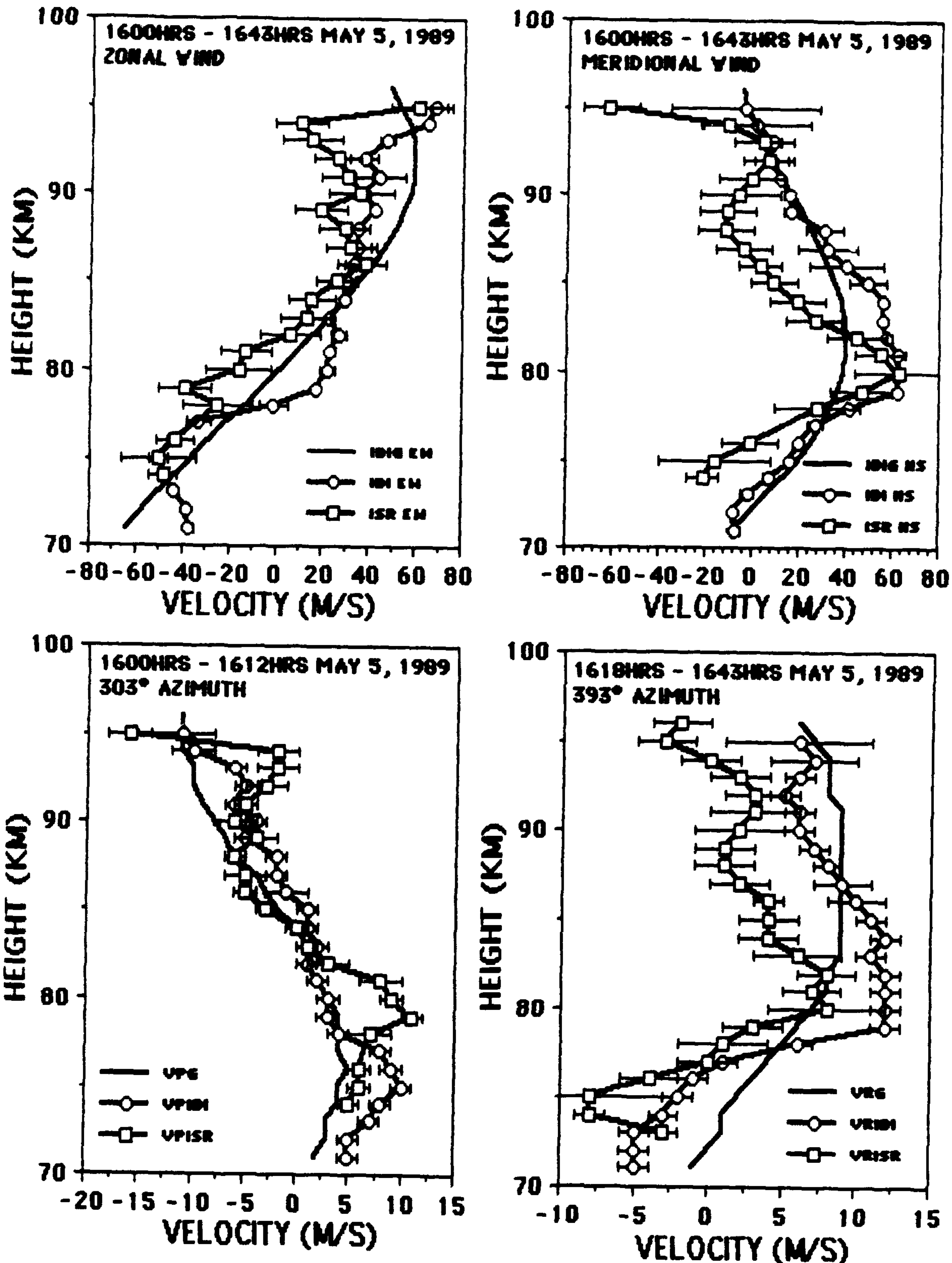

1600HRS - 1643HRS MAY 5, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1600HRS - 1643HRS MAY 5, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1600HRS - 1612HRS MAY 5, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1618HRS - 1643HRS MAY 5, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 19

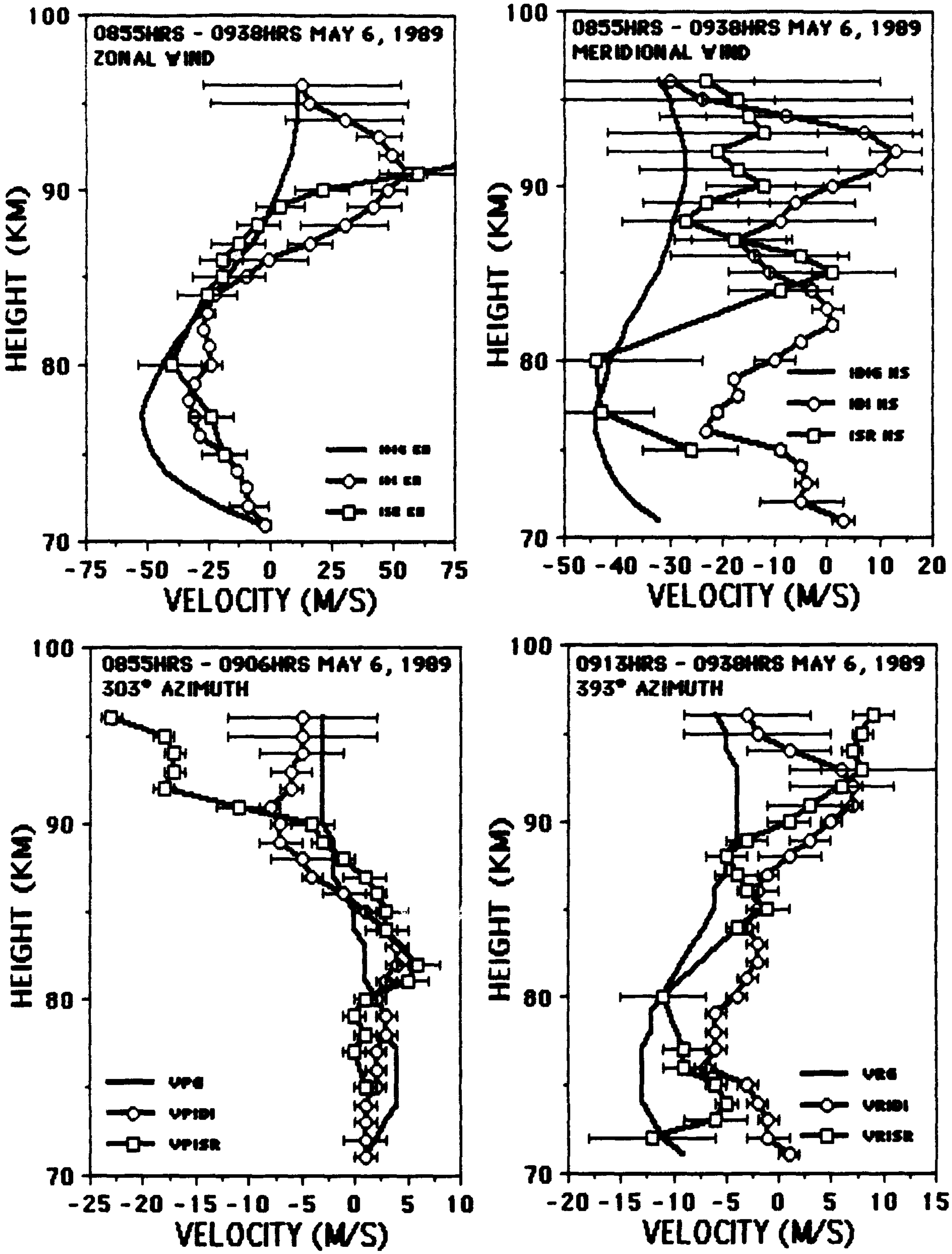

0855HRS - 0938HRS MAY 6, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0855HRS - 0938HRS MAY 6, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0855HRS - 0906HRS MAY 6, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
0913HRS - 0938HRS MAY 6, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 20

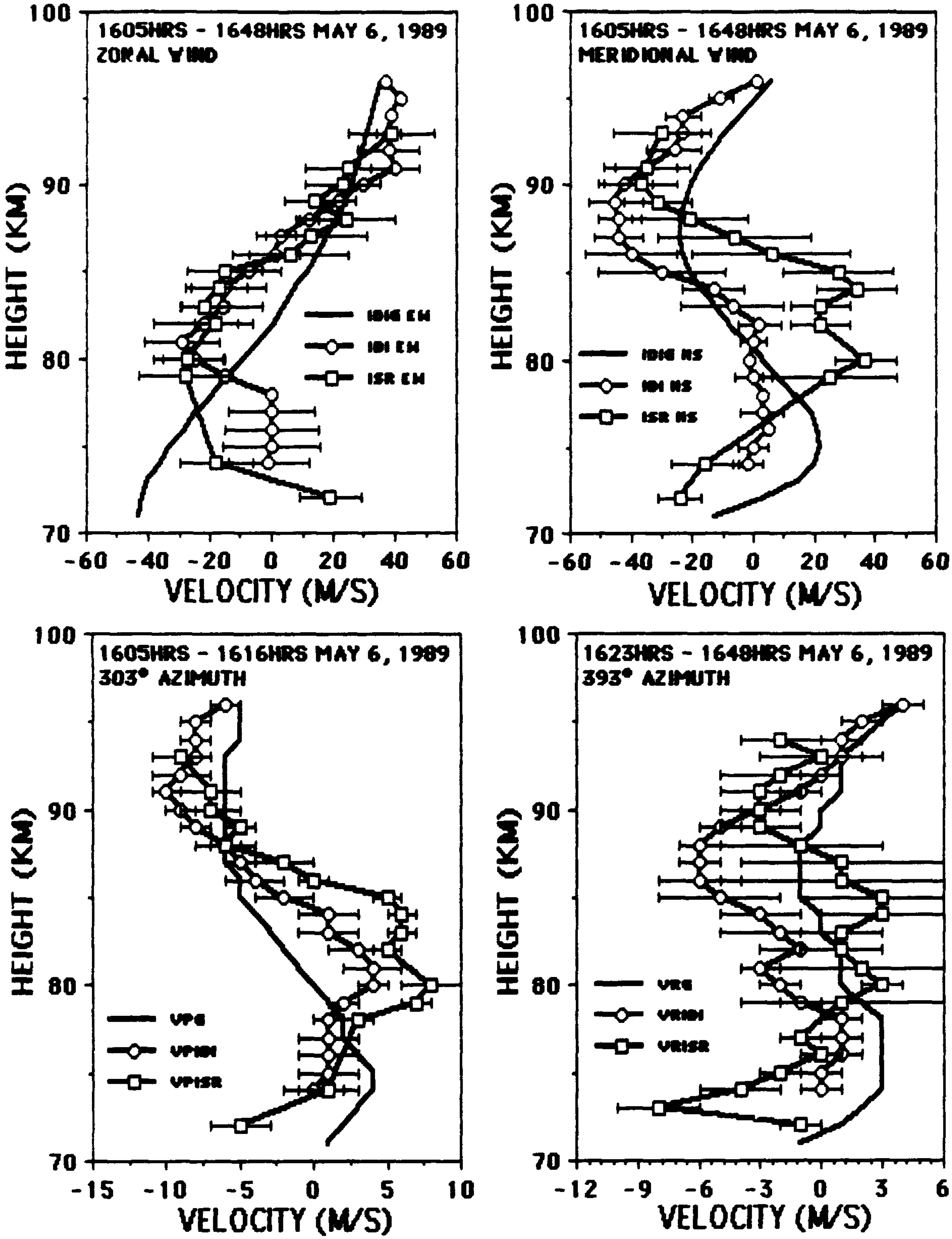

1605HRS - 1648HRS MAY 6, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1605HRS - 1648HRS MAY 6, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1605HRS - 1616HRS MAY 6, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
1623HRS - 1648HRS MAY 6, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 21

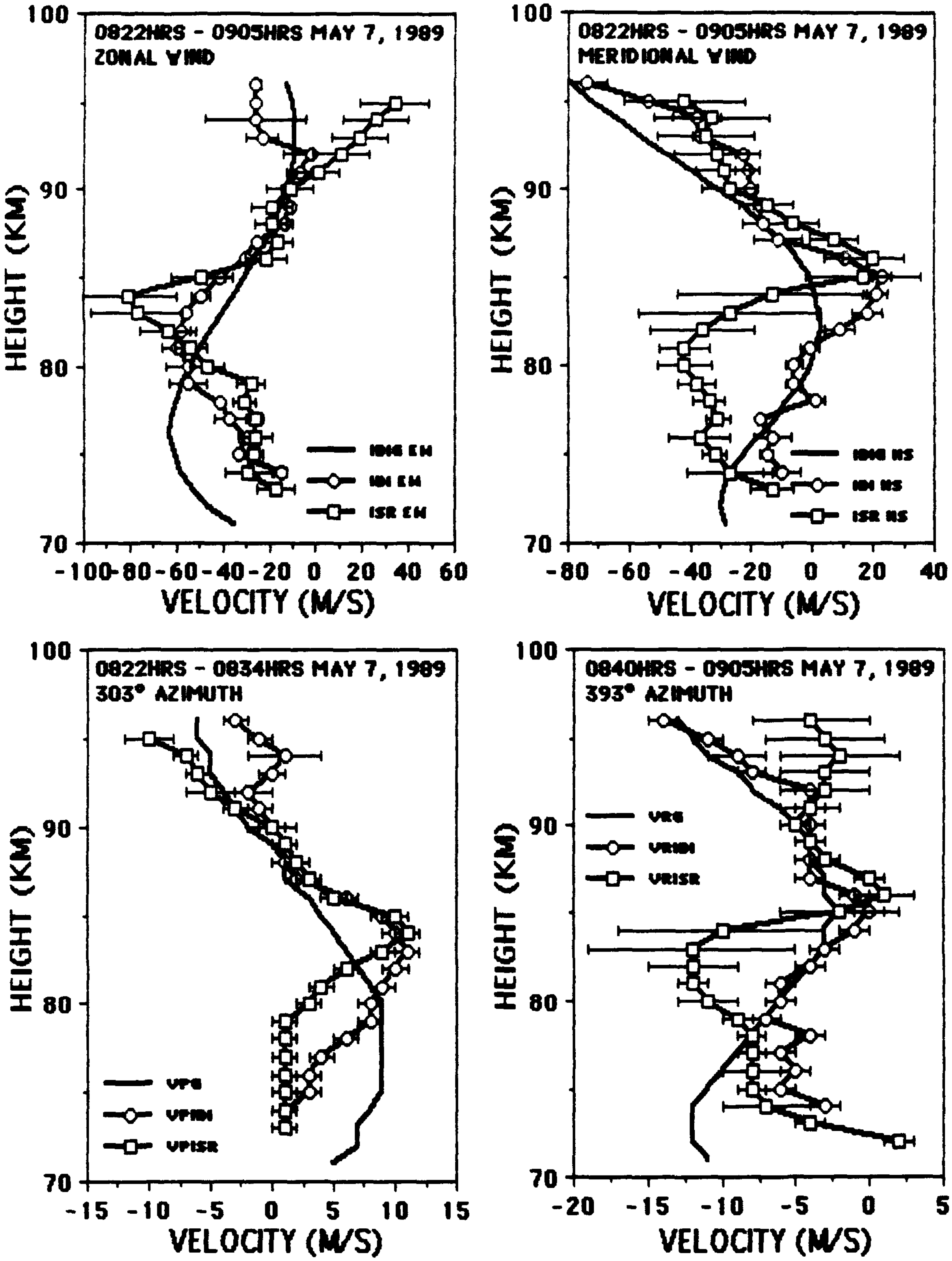

0822HRS - 0905HRS MAY 7, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
ISR EW
0822HRS - 0905HRS MAY 7, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
ISR NS
0822HRS - 0834HRS MAY 7, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VPISR
0840HRS - 0905HRS MAY 7, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
VRISR

FIGURE 22

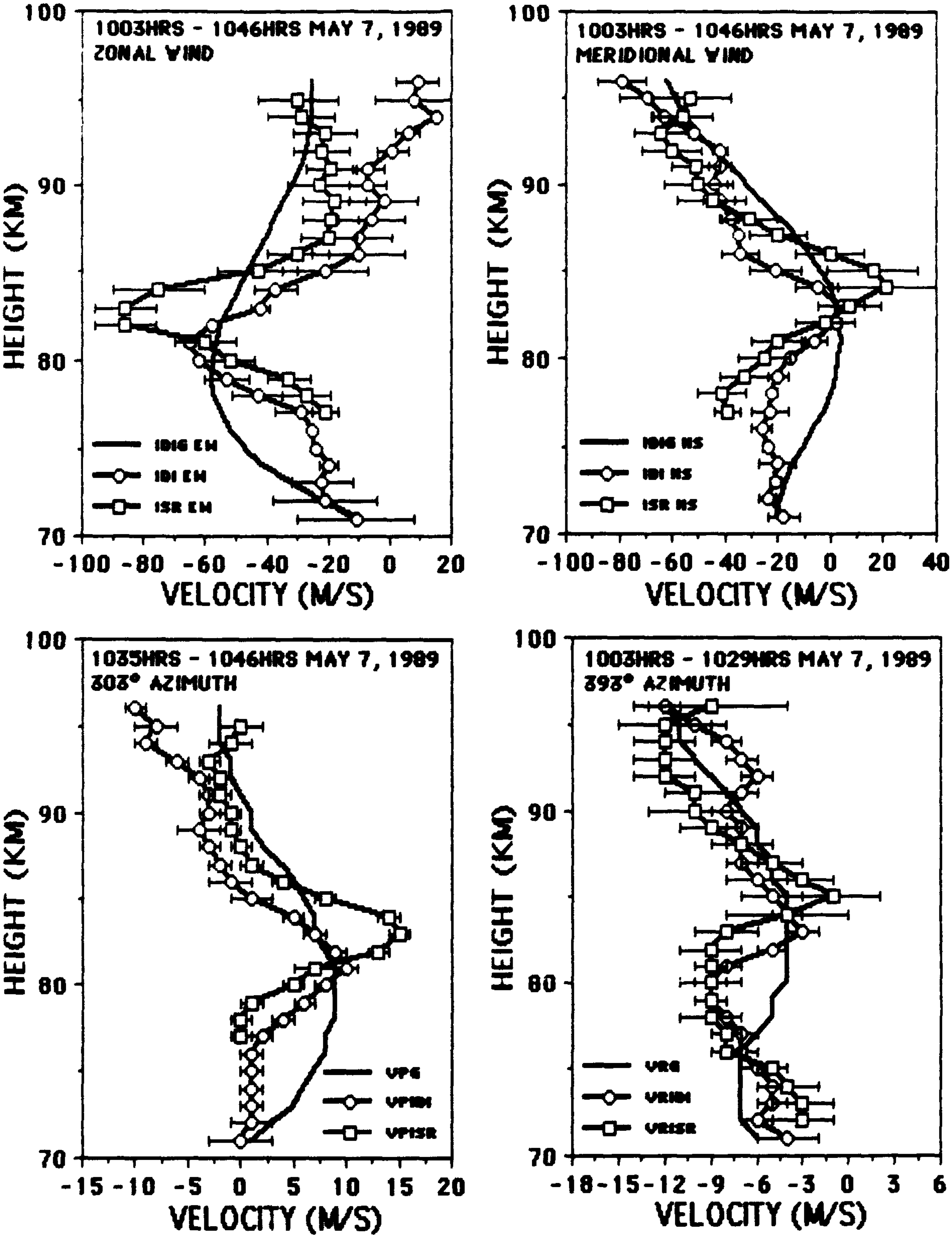
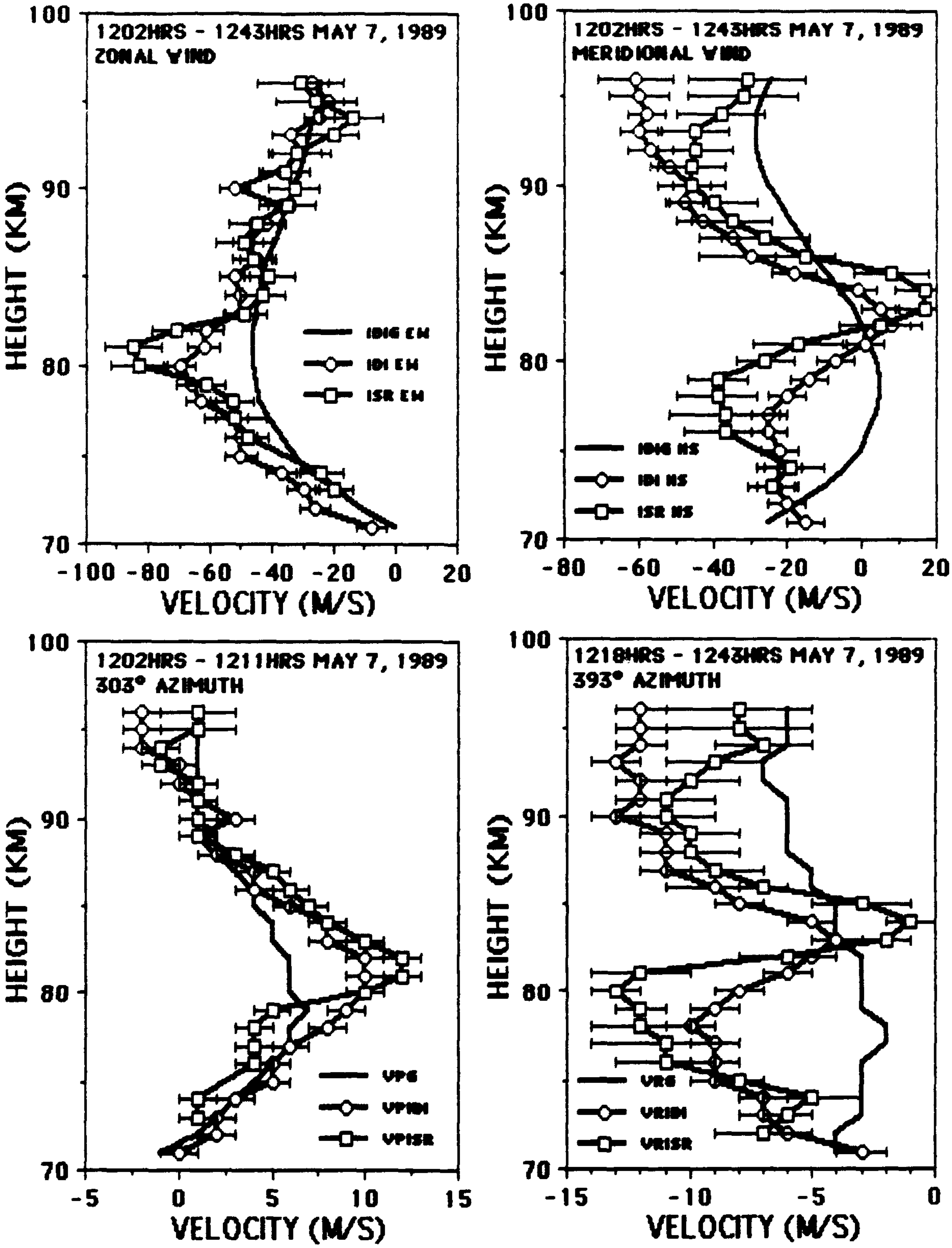

1003HRS - 1046HRS MAY 7, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1003HRS - 1046HRS MAY 7, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1035HRS - 1046HRS MAY 7, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1003HRS - 1029HRS MAY 7, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 23



1202HRS - 1243HRS MAY 7, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1202HRS - 1243HRS MAY 7, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1202HRS - 1211HRS MAY 7, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1218HRS - 1243HRS MAY 7, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)


FIGURE 24

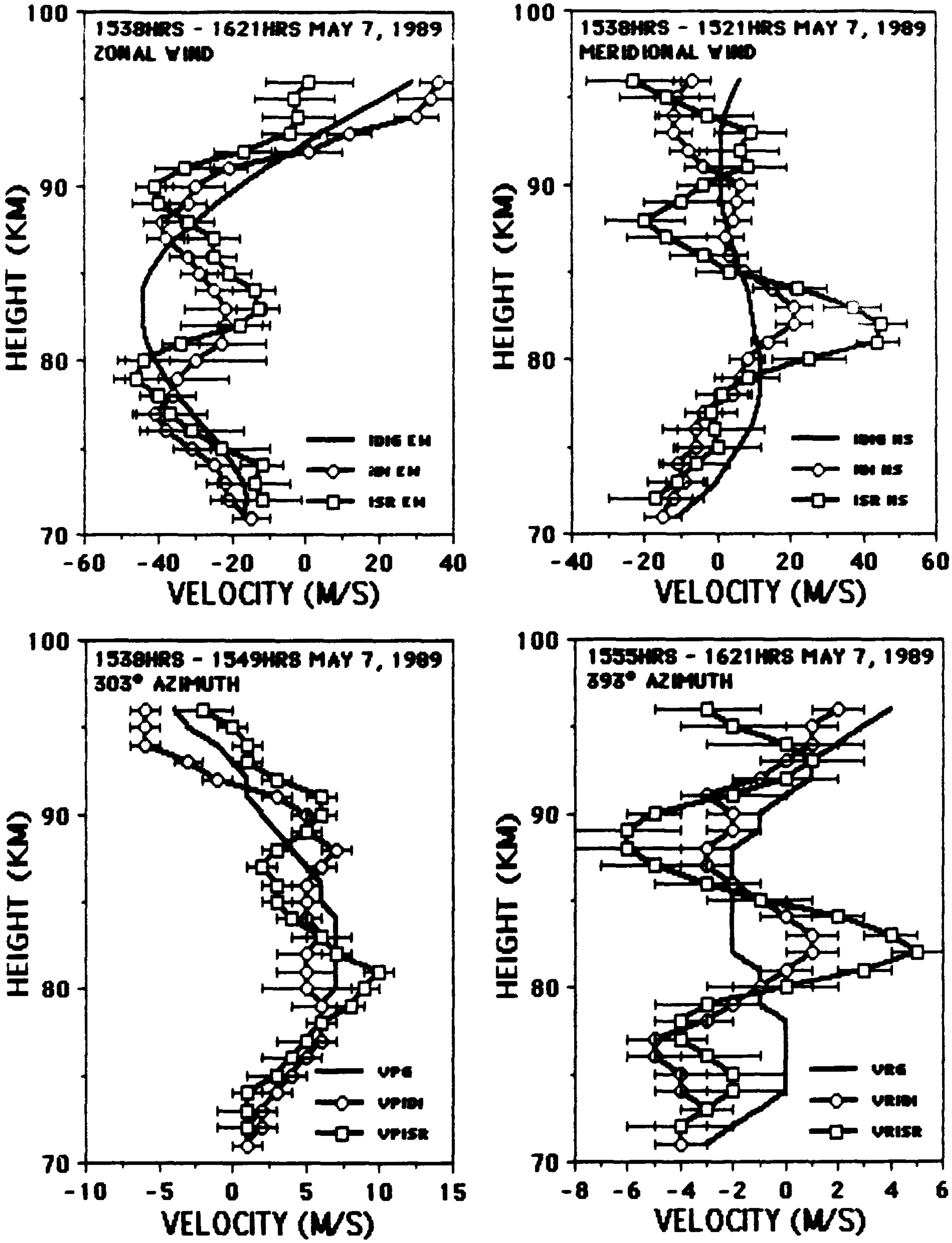

1538HRS - 1621HRS MAY 7, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1538HRS - 1521HRS MAY 7, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1538HRS - 1549HRS MAY 7, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1555HRS - 1621HRS MAY 7, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 25

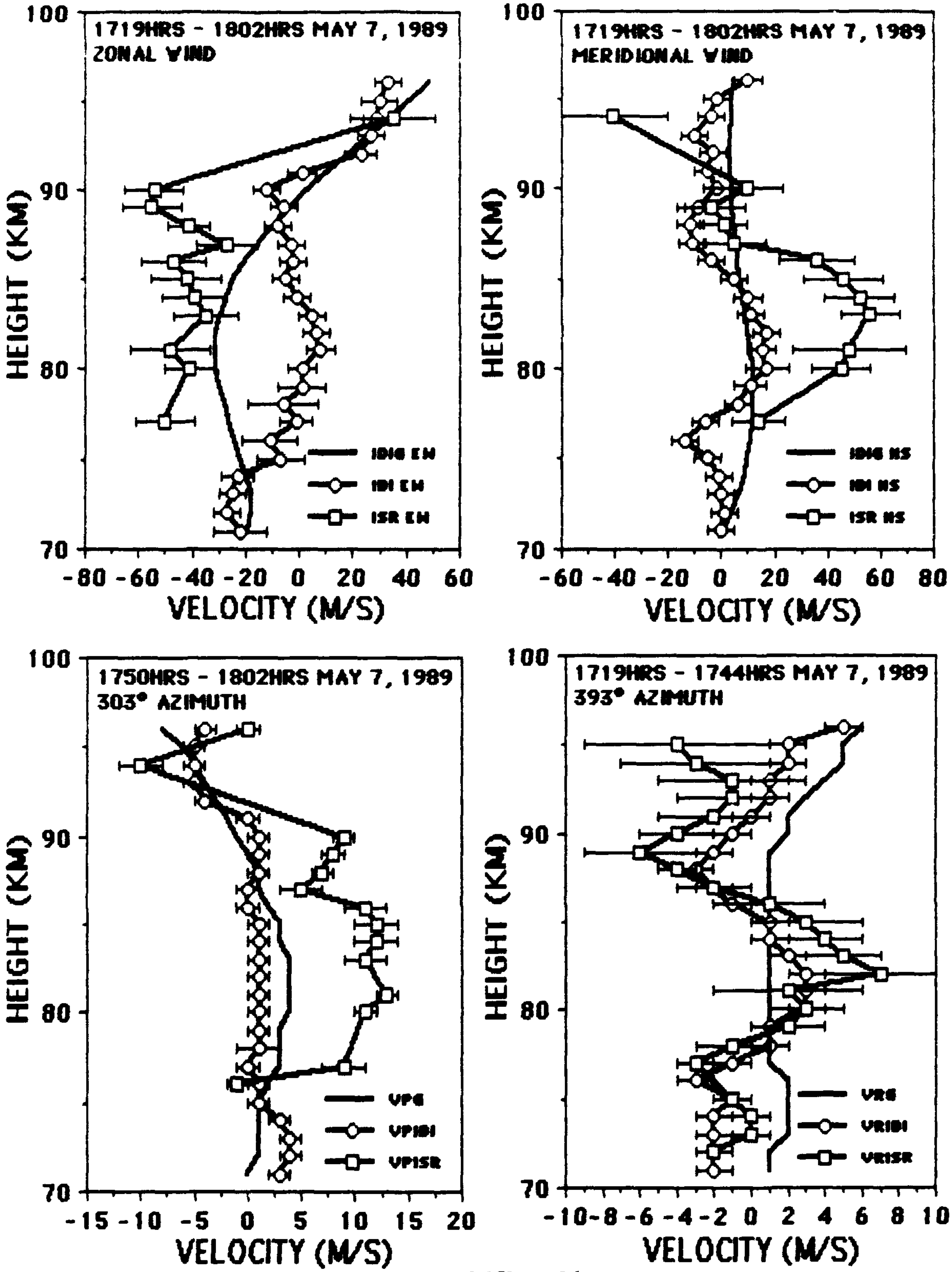

1719HRS - 1802HRS MAY 7, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1719HRS - 1802HRS MAY 7, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1750HRS - 1802HRS MAY 7, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1719HRS - 1744HRS MAY 7, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 26

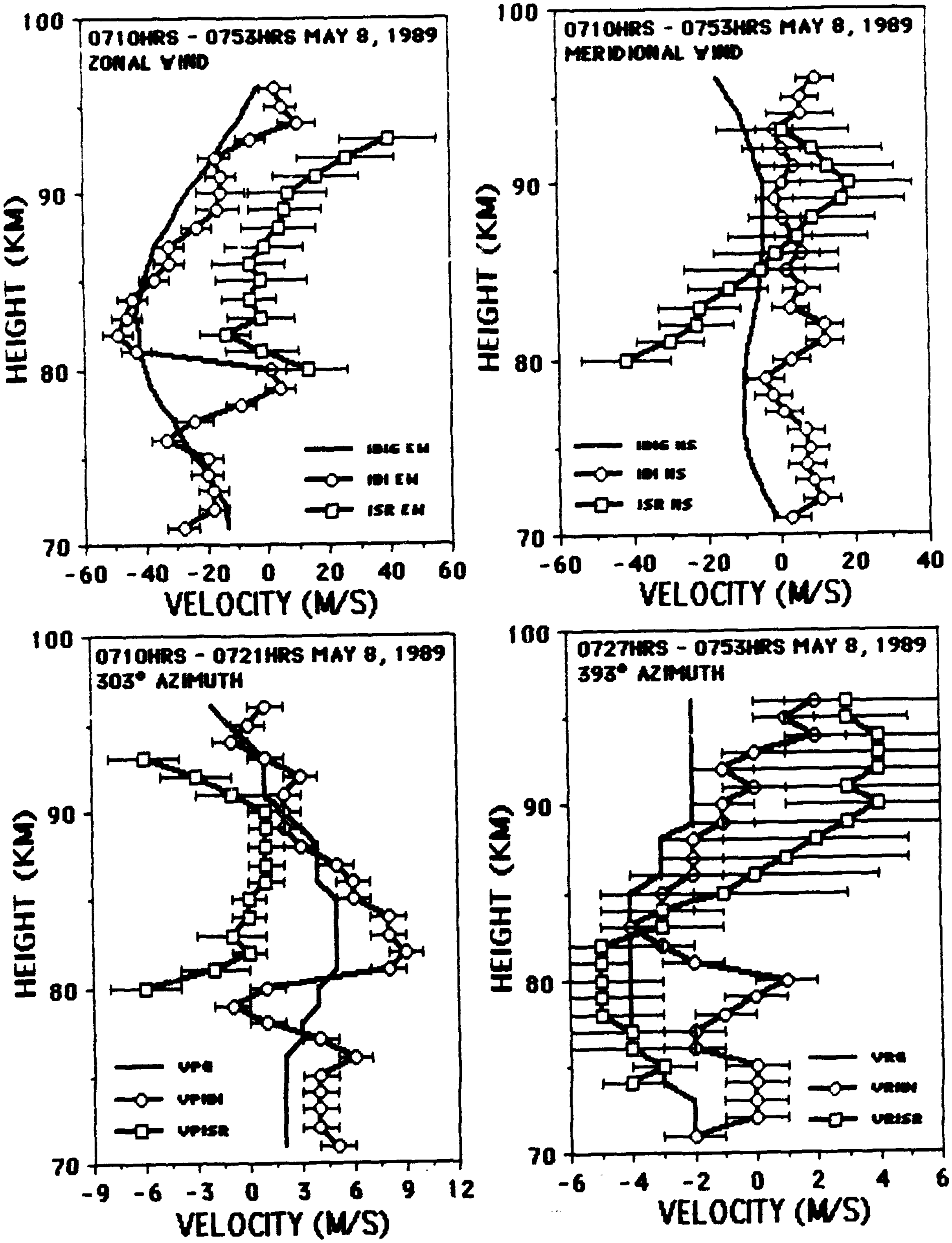

0710HRS - 0753HRS MAY 8, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0710HRS - 0753HRS MAY 8, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
0710HRS - 0721HRS MAY 8, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
0727HRS - 0753HRS MAY 8, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 27

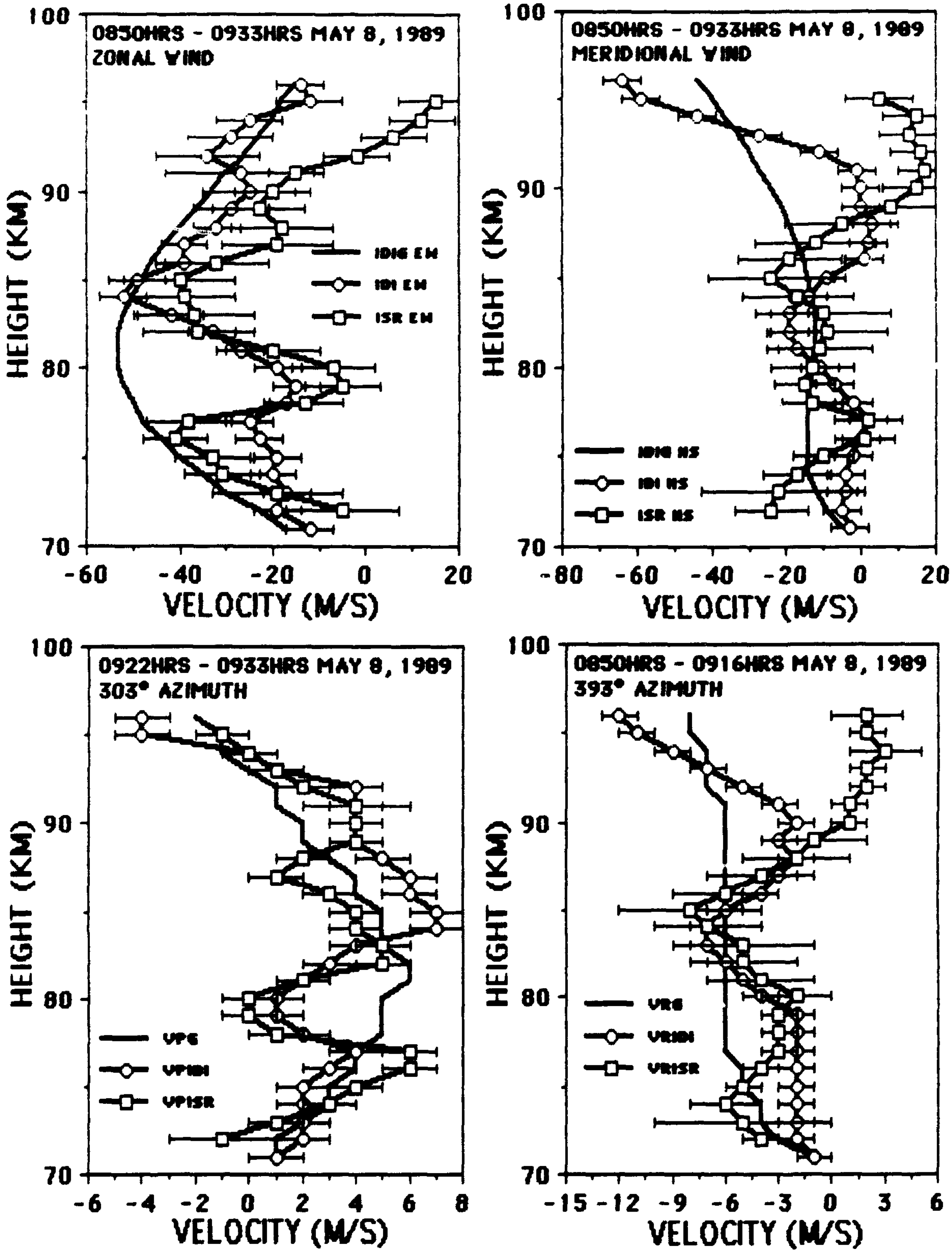

0850HRS - 0933HRS MAY 8, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
0850HRS - 0933HRS MAY 8, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
0922HRS - 0933HRS MAY 8, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
0850HRS - 0916HRS MAY 8, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 28

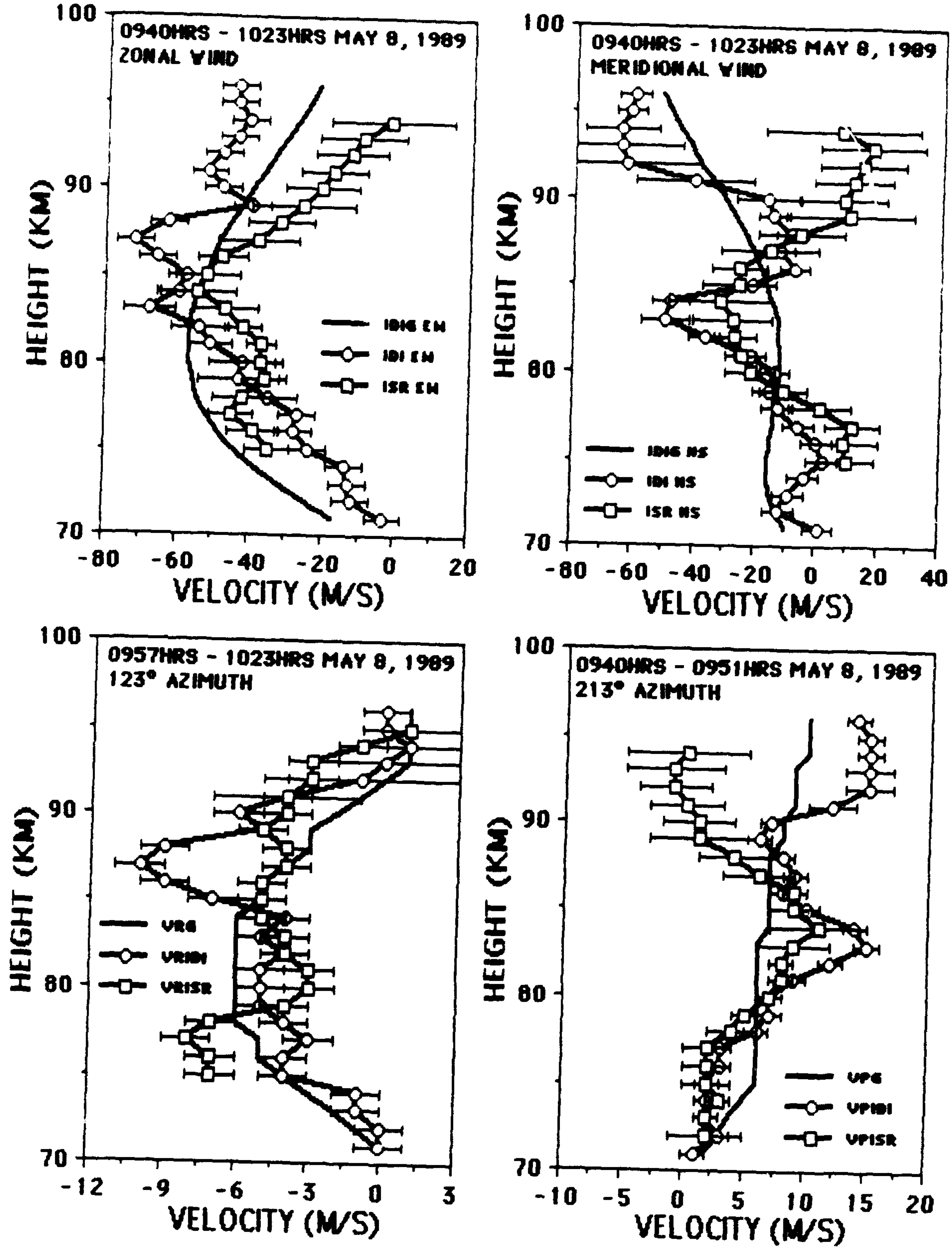

0940HRS - 1023HRS MAY 8, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
0940HRS - 1023HRS MAY 8, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
0957HRS - 1023HRS MAY 8, 1989
123° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)
0940HRS - 0951HRS MAY 8, 1989
213° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 29

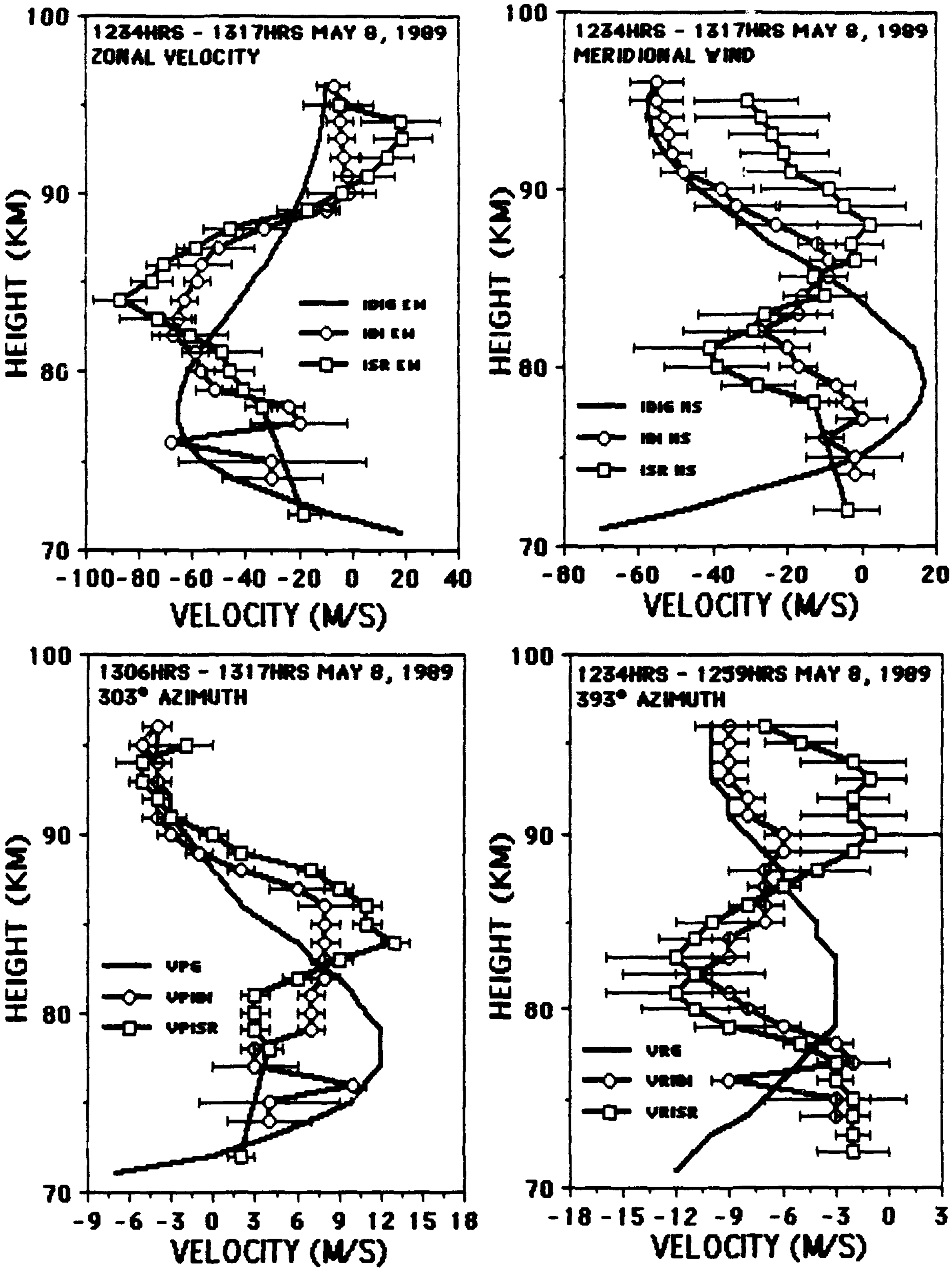

1234HRS - 1317HRS MAY 8, 1989
ZONAL VELOCITY
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
1234HRS - 1317HRS MAY 8, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
1306HRS - 1317HRS MAY 8, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
1234HRS - 1259HRS MAY 8, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 30

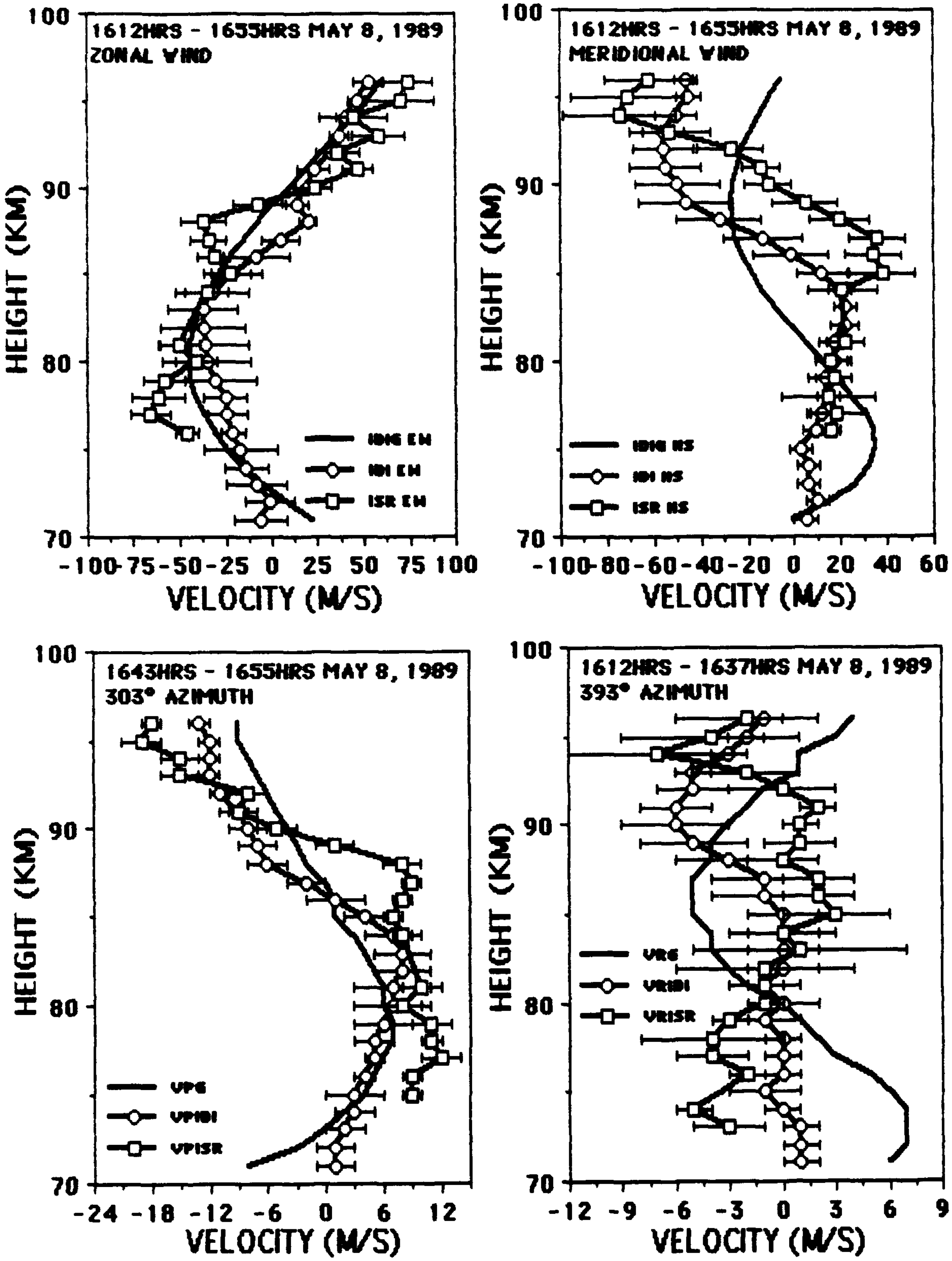

1612HRS - 1655HRS MAY 8, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
1612HRS - 1655HRS MAY 8, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
1643HRS - 1655HRS MAY 8, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
1612HRS - 1637HRS MAY 8, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 31

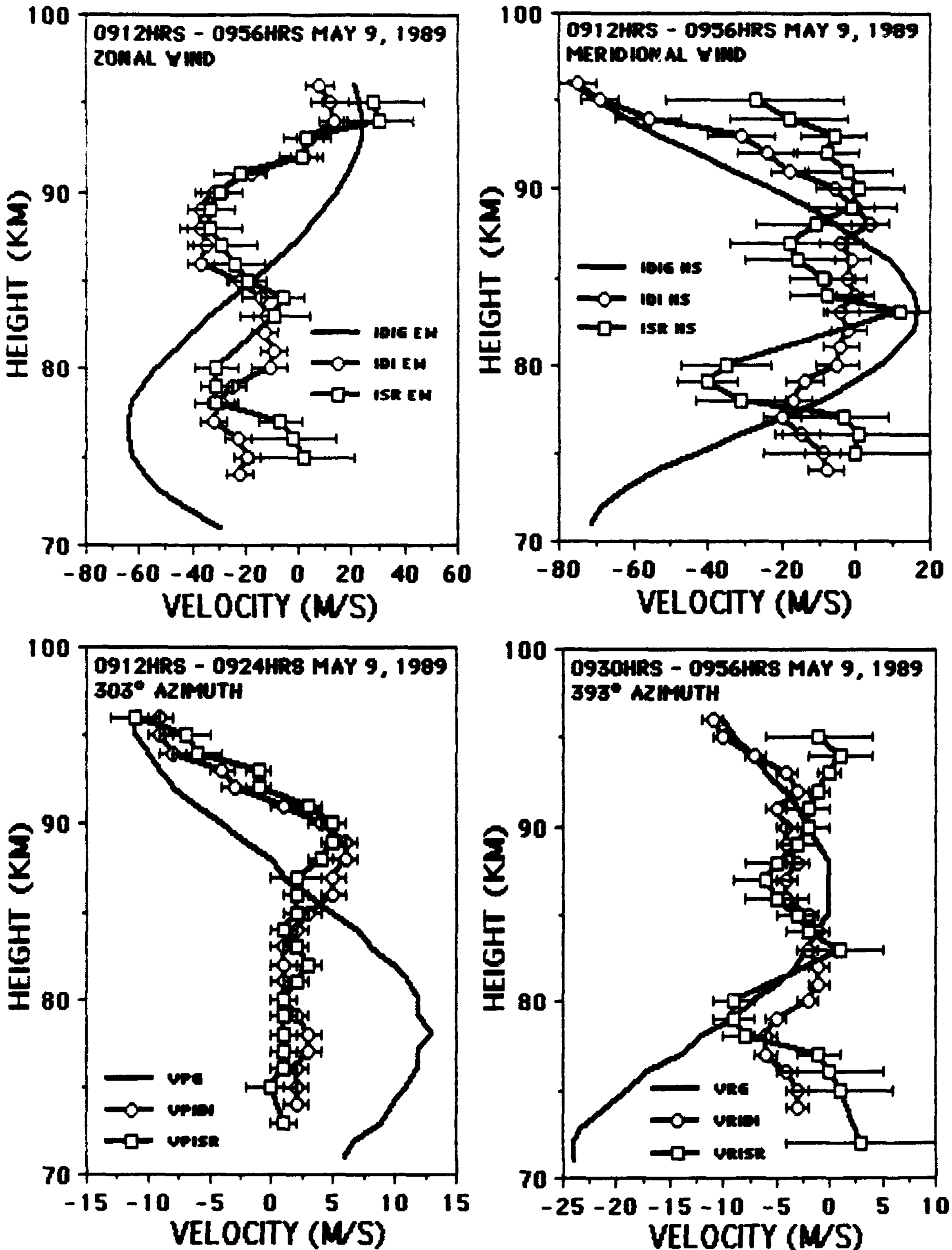
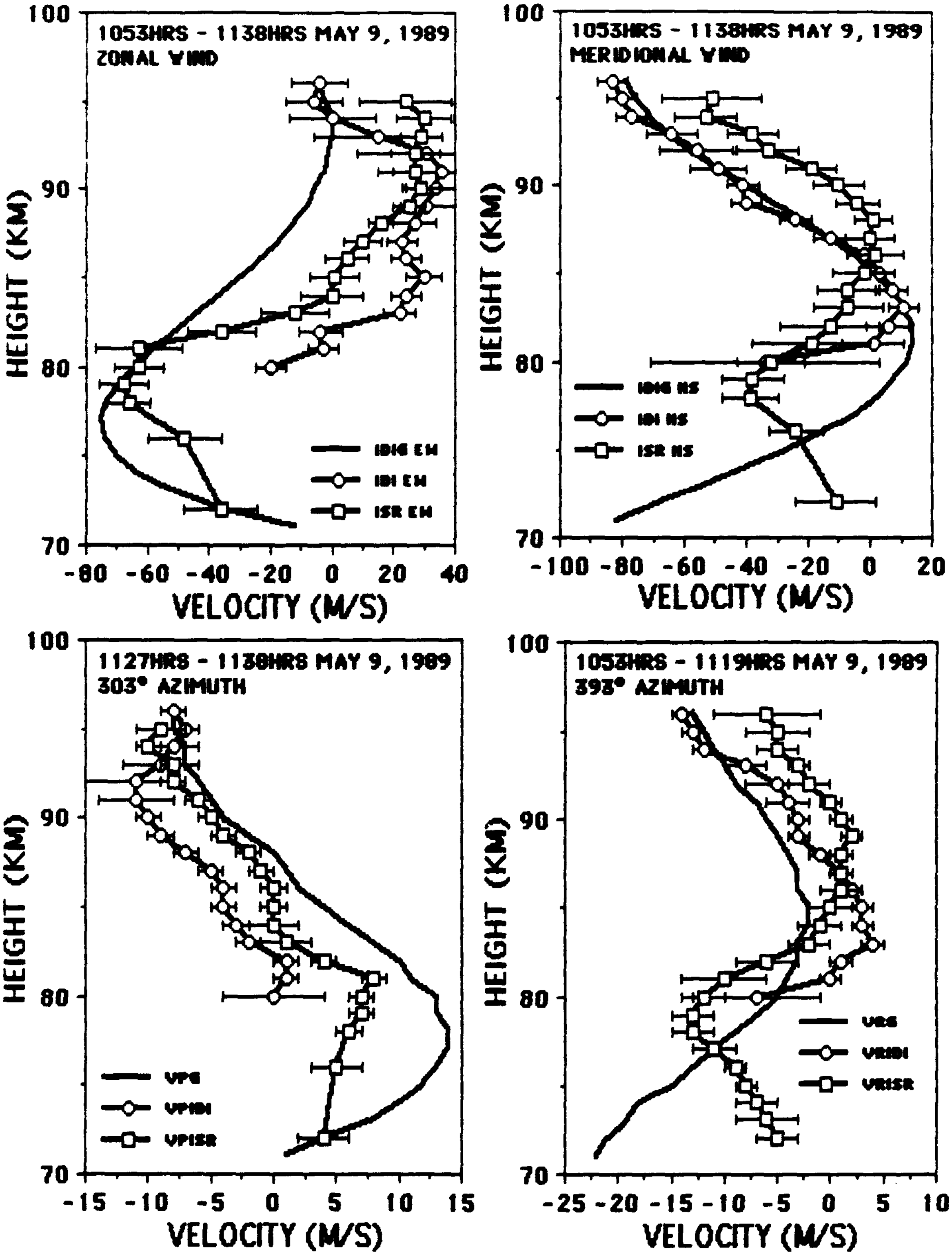

0912HRS - 0956HRS MAY 9, 1989
ZONAL WIND
IDIG EW
IDI EW
ISR EW
HEIGHT (KM)
VELOCITY (M/S)
0912HRS - 0956HRS MAY 9, 1989
MERIDIONAL WIND
IDIG NS
IDI NS
ISR NS
HEIGHT (KM)
VELOCITY (M/S)
0912HRS - 0924HRS MAY 9, 1989
303° AZIMUTH
VPG
VPIDI
VPISR
HEIGHT (KM)
VELOCITY (M/S)
0930HRS - 0956HRS MAY 9, 1989
393° AZIMUTH
VRG
VRIDI
VRISR
HEIGHT (KM)
VELOCITY (M/S)

FIGURE 32



1053HRS - 1138HRS MAY 9, 1989
ZONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1053HRS - 1138HRS MAY 9, 1989
MERIDIONAL WIND
HEIGHT (KM)
VELOCITY (M/S)
1127HRS - 1138HRS MAY 9, 1989
303° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)
1053HRS - 1119HRS MAY 9, 1989
393° AZIMUTH
HEIGHT (KM)
VELOCITY (M/S)


FIGURE 33

## IDI - FPS Comparisons

In this section, we present a preliminary look at six additional IDI - FPS comparisons not included in the Hines et al. (1993) paper,. The FPS data used here is from the Arecibo Radio Observatory Fabry-Perot Spectrometer as published in Bird et al. (1993).

In contrast to Hines et al., who compared the FPS data with the IDI winds at 94, 97 and 100km, we follow the method of Hernandez and Roper (1979), who, in comparing meteor radar and FPS winds, smoothed the meteor radar wind profiles with a green line profile to produce height averaged wind values. The smoothing profile used here weights the IDI data from 93 to 102km with the function graphed in Figure 34 (a gaussian of 7km half-width centered on 97km).

The FPS data was collected by stepping the spectrometer around eight cardinal points whose intersection with the 97km altitude level defined a circle of radius 170km, across which the line of sight drifts are averaged (allowing for a linear gradient with separation) to produce a horizontal wind vector every 21 minutes. The IDI scattering point parameter data (which are confined by rhe transmitter beamwidth to a circle of radius 20km at 97km) were analysed in 21 minute segments to produce altitude profiles which were then smoothed with the green line weighting function. We have not used the Hines et al. "discrepancy line" representation, but rather have shown our results in Figures 35 through 37 as zonal and meridional wind plots. We have also plotted the green line smoothed Groves winds (prevailing plus diurnal plus semidiurnal fits to the IDI data) for comparison.

Again, we have spent little time on the interpretation of these results. However, the IDI and FPS winds do show better agreement in the meridional than they do in the zonal (see Figure 38), which is just the opposite of the IDI - ISR comparisons! Of interest also is the considerably better agreemen t between the IDI and FPS winds after 0030 hours. The velocity differences are a factor of two smaller after 0030hrs than before (see, again, Figure 38). One might speculate, given the high shear with height of the IDI winds, that the green line maximum emmission altitude is not constant, at least before midnight. The solution is not that simple, however. The results from the night of May 3 - 4 (Figure 36) compare well throughout the night in the meridional component, and in the zonal component after midnight, but disagree by some 100m/s in the zonal at 20 - 2100hours! Obviously, these results warrant further investigation.

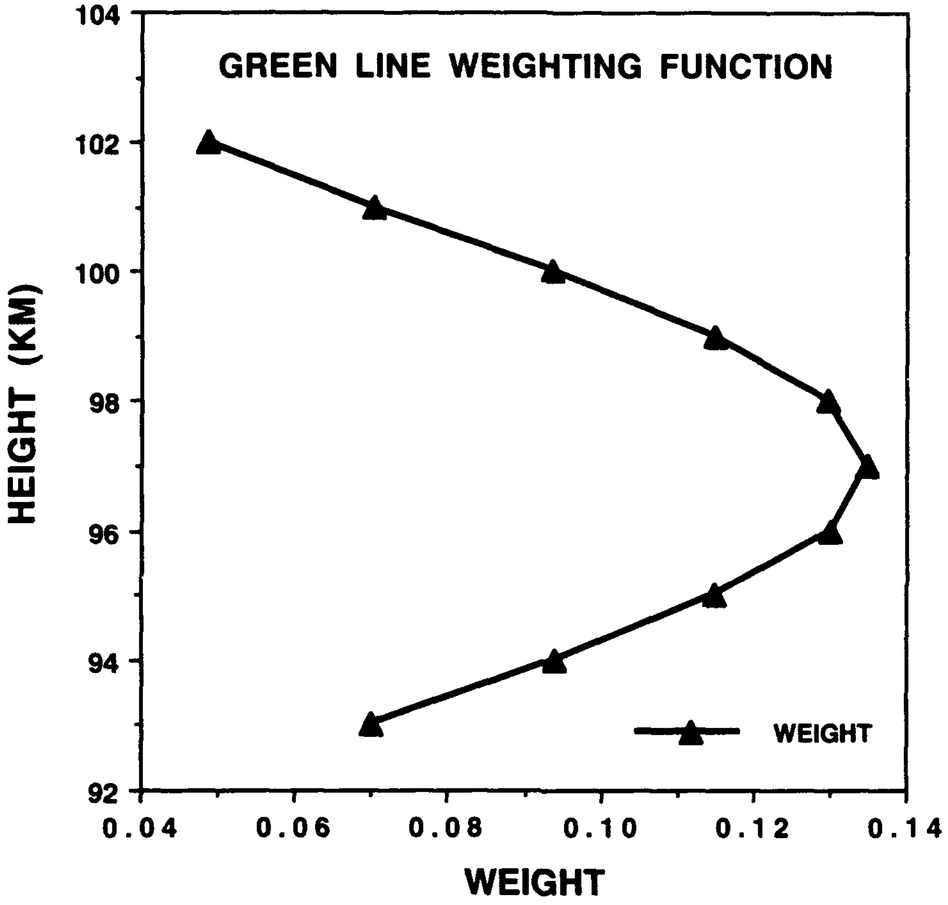

GREEN LINE WEIGHTING FUNCTION
HEIGHT (KM)
104
102
100
98
96
94
92
0.04
0.06
0.08
0.10
0.12
0.14
WEIGHT
WEIGHT

FIGURE 34

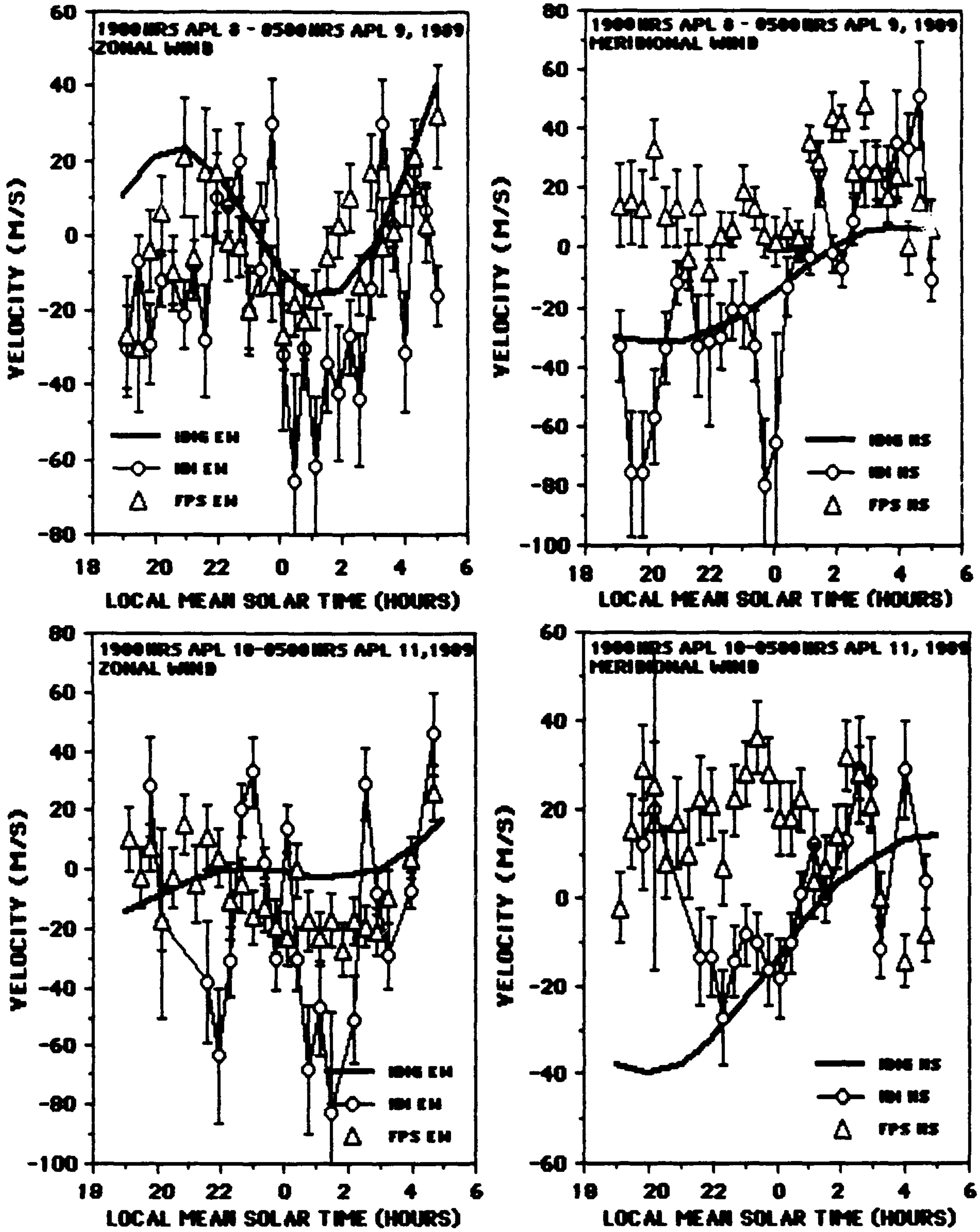

1900HRS APL 8 - 0500HRS APL 9, 1989
ZONAL WIND
1900HRS APL 8 - 0500HRS APL 9, 1989
MERIDIONAL WIND
1900HRS APL 10-0500HRS APL 11,1989
ZONAL WIND
1900HRS APL 10-0500HRS APL 11, 1989
MERIDIONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)

FIGURE 35

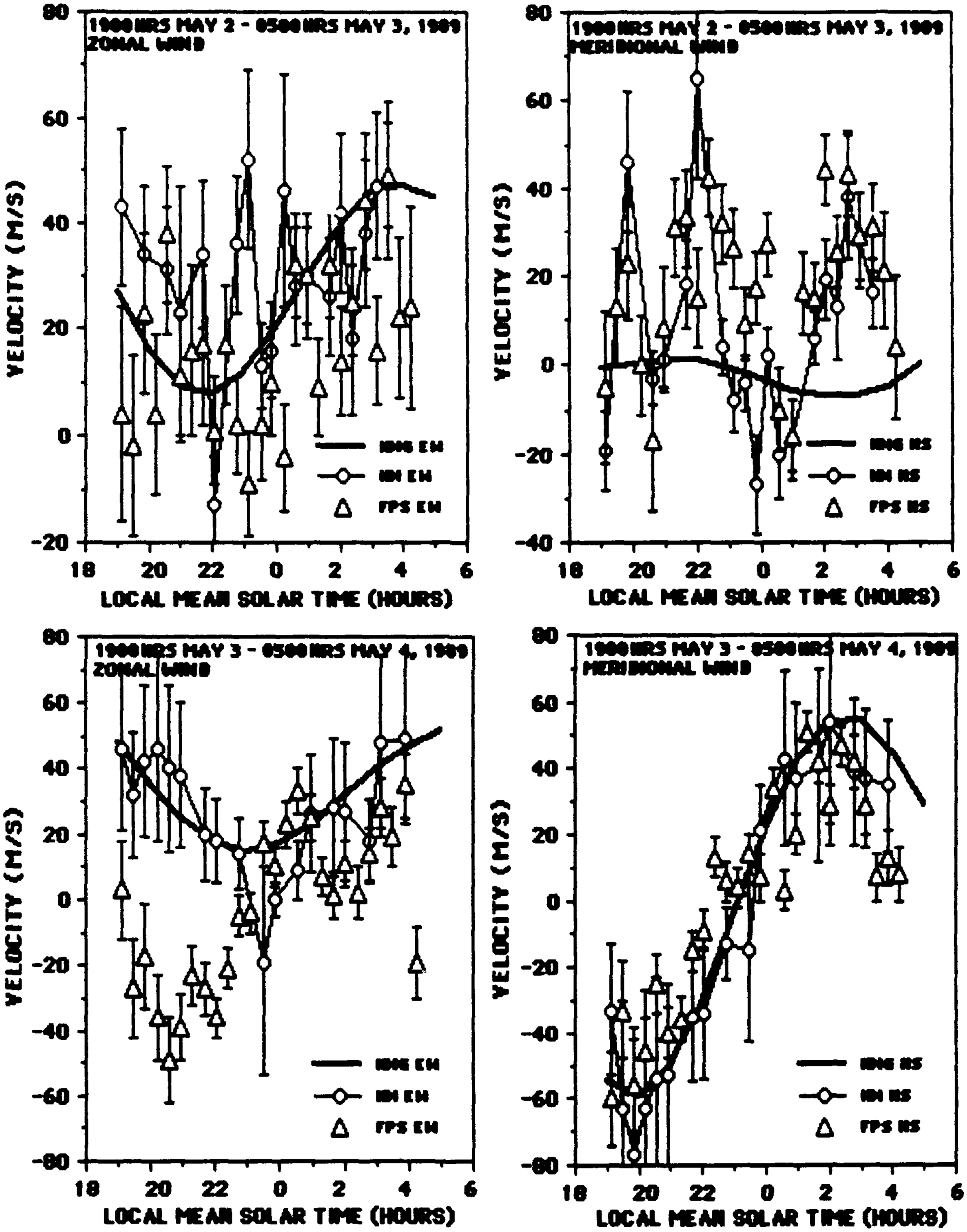

1900 HRS MAY 2 - 0500 HRS MAY 3, 1989
ZONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS EW
1900 HRS MAY 2 - 0500 HRS MAY 3, 1989
MERIDIONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS NS
1900 HRS MAY 3 - 0500 HRS MAY 4, 1989
ZONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS EW
1900 HRS MAY 3 - 0500 HRS MAY 4, 1989
MERIDIONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS NS

FIGURE 36

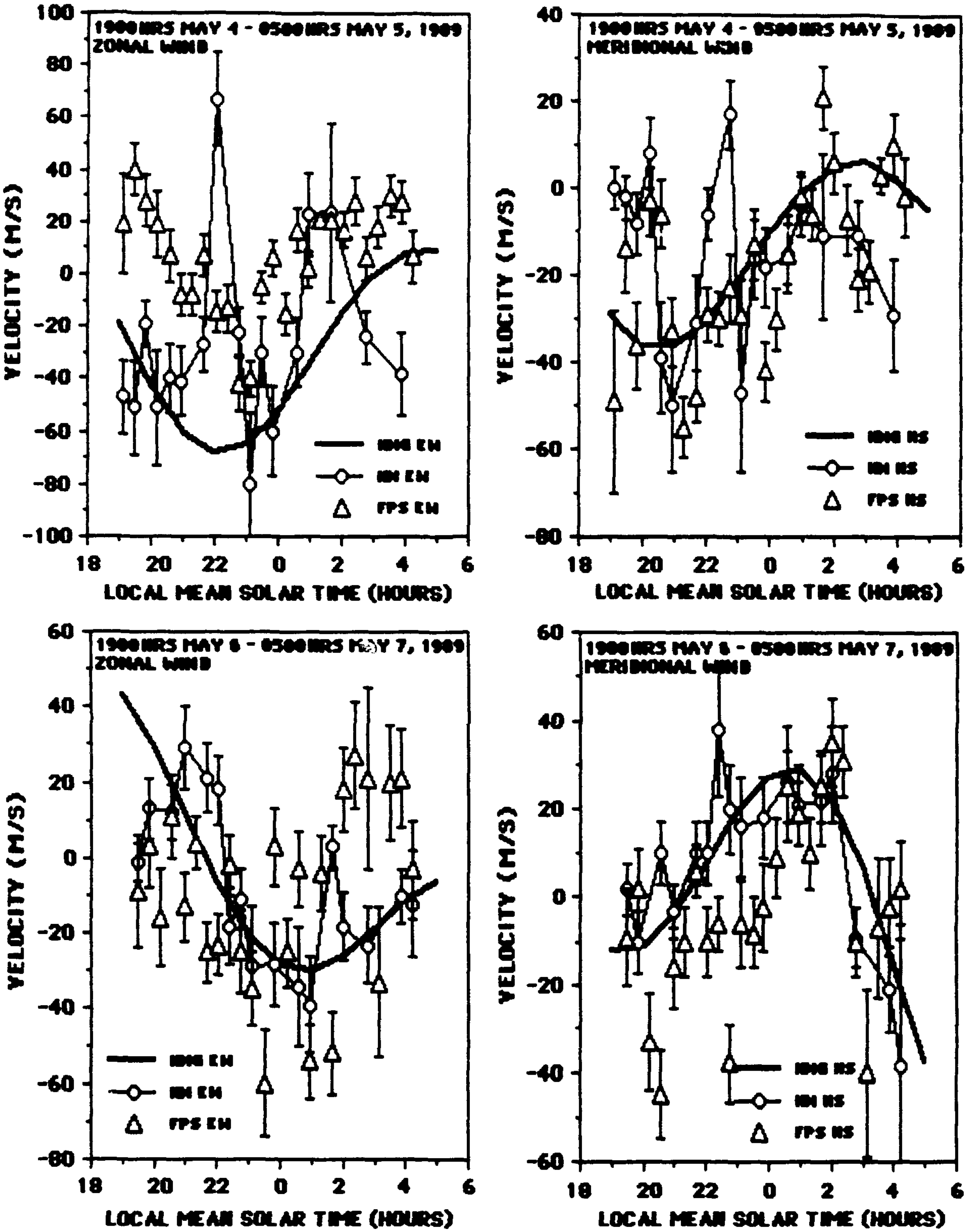

1900 HRS MAY 4 - 0500 HRS MAY 5, 1989
ZONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS EW
1900 HRS MAY 4 - 0500 HRS MAY 5, 1989
MERIDIONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS NS
ZONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS EW
MERIDIONAL WIND
VELOCITY (M/S)
LOCAL MEAN SOLAR TIME (HOURS)
FPS NS

FIGURE 37

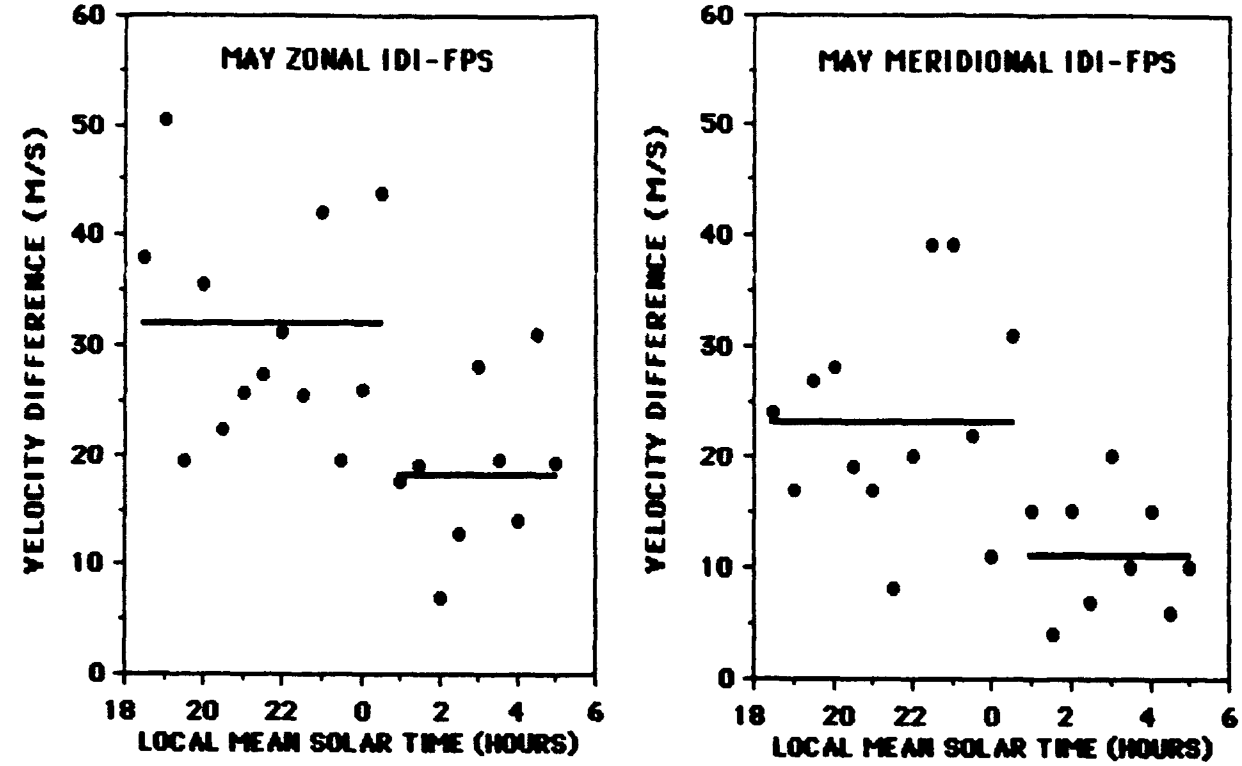



FIGURE 38

## References


Bird, J.C., G.G. Shepherd and C.A. Tepley, "Comparison of lower thermospheric winds measured by a Polarizing Michelson Interferometer and a Fabry-Perot spectrometer during the AIDA campaign.," J. Atmos. Terrest. Phys., 55, 313-324, 1993

Groves, G.V., "A theory for determinimg upper atmospheric winds from radio observations of meteor trails," J. Atmos. Terrest. Phys., 16, 344-356, 1959

Hernandez, G and R.G. Roper," A comparison between radio meteor and airglow winds," J. Geomag. Geoelect., 31, 419-, 1979.

Hines, C.O., G.W. Adams, J.W. Brosnahan F.T. Djuth, M.P. Sulzer, C.A. Tepley and J.S. Van Baelen, "Multi-instrument observations of mesospheric motions over Arecibo: comparisons and interpretations", J. Atmos. Terrest. Phys., 55, 241-288, 1993

Roper, R.G., G.W. Adams and J.W. Brosnahan, "Tidal Winds at mesopause altitudes over Arecibo (18°N, 67°W), 5 - 11 April, 1989 (AIDA'89)," J. Atmos. Terrest. Phys., 55, 289-312, 1993

Turek, R. Scott, K.L. Miller, G.W. Adams, R.G. Roper and J.W. Brosnahan, "Mesospheric wind studies during AIDA Act '89: morphology and comparison of various techniques," in preparation.

## Part 2 - The MAPSTAR Imaging Doppler Interferometry (IDI) radar Data Reduction and Comparative Analysis Computer Programs.

This second part of this two part report outlines the procedures to produce the data tables and graphs presented in Part 1.

The first section deals with the reduction and analysis procedures developed by Gene Adams and used by him and his students to reduce and analyse the MAPSTAR radar data at Utah State University. Input data are the digitized raw signal tapes (for the purposes of this report, those recorded as the outputs of each of the MAPSTAR radar receivers during AIDA Scene III - May 2 - 9, 1989, but applicable to any interval). Data was recorded in this form so that it could be subsequently analysed using algorithms appropriate to other partial reflection interferometry and spaced antenna techniques. The results of such other analyses are part of currently proceeding analyses, and will be reported later. Here, we concentrate on the Imaging Doppler Interferometry (IDI) analysis, details of which may be found in Brosnahan and Adams (1993).These programs, written in IBM PC compatible FORTRAN 77, determine the individual scattering point parameters (time of occurrence, height, line of sight velocity, azimuth and zenith angles and polarization of return) and use these to determine hourly mean zonal, meridional and vertical wind profiles, with errors, over selected height ranges. A detailed writeup of these procedures, produced by Gene only two months before his death, follows.

process2.mem

To: Files

From: Gene W. Adams

Subject: Explanation of GR-AIDA Tape Processing

This documentation initially follows a 20-point check list I made for myself. This memo will, I hope, explain enough of what's going on to get you through it. There's more after the 20-point list, but it's not crucial (you could redo the software if you had to).

Processing a tape on an IBM-PC involves reading the original raw-data 9-track tape, separating and somehow storing the information in the tape and record headers and the associated strings of data, inverting the byte order of the data (program named GAFIX.for), putting the data through a 4-pass filter (program named FLTRB.for) to remove spikes, removing the dc components of the signal, adjusting all 10 channels to have common gain, and correcting the phases of the 10 channels. The "calibrated" data are stored on dat and erasable optical (EO) disks, and on 9-track tape as needed to satisfy requests from others that want to reduce our data with their algorithms (Joel Van Baelen and Erhan Kudeki, so far). The IDI algorithms are then used (program named BSPPM.for) to determine the scattering-point parameters (SPPs) from the calibrated data. The bulk of the processing sequence can be done from a batch file (batch file named MOJO.bat), but manual intervention is still needed to identify tape-write errors, enter the tape number, etc. That's why my check list is long.

The quoted command at each number is from my 20-point check list:

1. "Boot up with \final disk in EO drive." An erasable-optical disk with a \final directory is where I collect the scattering-point parameter files for the tape just processed. They carry the time of each sounding (102.4sec/sounding) followed by the SPPs, ordered by altitude, lowest altitude first.

2. "Open and close Windows." Opening and closing Windows 3.0 seems to be required on my system to get the cursor to go fast. Otherwise, it's slow.

3. "Delete all files in \Buffer01 and \Buffer02." \Buffer01 and \Buffer02 are two subdirectories on the C: drive, each big enough to hold a tape's worth of data. Since the filter processes an entire tape at once, the tape in its various stages of processing are written from \Buffer01 to \Buffer02, back to \Buffer01, etc., until it's done. It starts with the raw data in \Buffer01, and finishes with the calibrated data in \Buffer02.

4. "Mound and load raw data tape on 9-track drive." An original data tape should never be read but once. They're far too precious and irreplaceable to do anything but immediately make a back-up. This is where we make the back-up.

5. "Put up post-it with tape number on it." The tape number (e.g., GR-235) is not automatically entered anywhere, so keep careful track and you can enter it in the appropriate place below.

6. "Run TapetoC.bat, which read tape files into c:\Buffer01." A copy of TapetoC.bat is attached. It's pretty simple, and names the files as read in as 1.mbr, 2.mbr, etc. The extensions are used to label the type of file. The first letter is m for medium frequency (the MAPSTAR radar) or v for vhf (the MENTOR radar). The second letter is b for binary or a for ascii. The third letter is r for raw data, t for time-domain-average (calibrated) data, etc.

7. "Invoke DAT software with C:>tpu -<return>, then FSFnn at the ~ prompt to advance DAT nn files to end." This assumes that you're putting the calibrated data tapes on a DAT tape, and that you've already got some files on the tape. This command just moves the DAT tape to the end of the nn files you've already got, and leaves it ready to record the next tape's worth of files (there's usually 46-50 files per tape).

8. "Examine \Buffer01; note odd-length files." If there was a tape-write error during the radar operation, you'll get a file that's longer than all the others, because two files get transmogrified into one. You want to skip these files, and can erase them if it seems better.

9. "Edit FixTape.bat to skip odd-length raw-data files." FixTape.bat (attached) invokes the fortran program GAFix.for (copy attached, along with subroutines RdHdr.for, Names.for, and WrHdr.for) to separate the data from the ascii header information, and write the byte-inverted data to \Buffer02 (which can be changed in the source code, where it is named "pathout".) FixTape.bat can be made to skip file number 29, for instance, by just putting a "goto 30" right before :29. Notice that it uses drive E:, which is my ram drive. Change this to whatever drive you have available so that GAFix has a place to work (it needs room for 2 copies of a single data file: less than 2 Mbytes.

10. "Edit MOJO.bat to write properly named files." MOJO.bat, in step number 5, names the list of sounding times (contianed in gafix.txt) and the SPPs (contianed in bsppm.mbs) according to the tape number (the GR number). It is entered manually at this point by editing MOJO.bat.

11. "Fill up paper tray." It takes about half a tray of paper if you're going to image the screen so you can tell if it all worked once you're done. It can make a catastrophic mistake and return you pure garbage, and if you haven't been monitoring the screen you probably won't know about it.

12. "Turn on screen copy." Cntrl-P toggles the print-screen command. Hit it again when you're all done.

13. "Run Mojo.bat (takes about 4hr 40min)."
13A. "Copy raw files from \Buffer01 to DAT (45min)." This gets a true back-up of the original data tape onto DAT. We will also save the calibrated tape, but just in case you ever want to change the calibration procedure...

13B. "Run FixTape.Bat". As explained above, this separates data from header info and inverts the byte order (if selected by a switch in the source code). An entire tape is processed, and the finished files written to \Buffer02. The file names, which carry the time of the sounding to the second, have been written to gafix.txt. The format is MMDDHHMM.SSt where MM = month, DD = day, HH = hour, MM = minute, SS = second, and t denotes calibrated time-domain data. Example: 04281345.17t. Time is corrected to WWV and to the center of the sounding, so time span is +/- 102.4/2sec around given time. The tape-header and sounding-header is read by GAFix, but is not handled by the software beyond this. You have to edit the header.dat file for each run. However, the time is kept by the name of the file, and generally nothing else changes, so this works out okay.

13C. "Run FltrB.exe." FltrB.for is attached, as are the subroutines FltrB1.for, FltrB2.for, FltrB3.for, and FltrB4.for. The batch file (Mojo.bat) copies the list of file names, which are also the times, from gafix.txt to FltrB.txt. FltrB.for just keeps the file-names straight and the buffers straight, and calls the four subroutines. FltrB1.for makes one pass through the tape to determine the dc average for each of the 20 channels, a second pass to determine the rms deviation of the averaage, and a third pass to recalculate the dc average excluding points whose deviation from the average is more than 3 sgima. A fourth pass is made to subtract the dc average from the data in each channel. FltrB2 removes noise burts, defined as single data points that are 20(?)dB or more above the running average. FltrB3 calculates the average power in each of the 20 channels. FltrB4 adjusts the signal strength in each of the 20 channels so that they all have the same average power, and the phases are adjusted relative to the x-quadrature channel of antenna/receiver #5, which is used as the phase reference). The phase corrections were determined by averaging over 2+ hours of solid daytime E region, and assuming that the echoes were, on the average, in the zenith. This clearly needs to be repeated and several episodes averaged to get better numbers.

13D. "Run Bsppm.exe." B is for batch and m is for medium-frequency. This will need rewriting when the MENTOR data comes in, but it's all in one subroutine and it's easy. The screen displays, one sounding at a time, the number of scattering points found at each altitude and the number rejected by the various criteria. The program is Bsppm.for, with subroutines BfftM.for (the FFT driver), FFT2cm.for (the FFT routine itself), Header (which reads

header.dat, an ascii file that contains all the radar settings, and which you construct out of the existing one edited to be accurate for whatever you're doing), BtestM.for (which applies the IDI algorithms to see if a particular spectral window at a particular altitude is a scattering point or not), BSteerM.for (which steers the array towards scattering points to get the best estimate of their amplitudes and phases), and BSortM.for (which orders the scattering points by altitude, ready for output). The SPPs are written in binary to a file called BSppM.mbs (m = medium-frequency; b = binary; s = scattering-poing parameters). Notice that this one .mbs file contains the SPPs for the entire tape; the format is a 10-numbered time line (first number is -999) followed by many 10-numbered lines of SPPs. These are: altitude, radial velocity, E-W zenith angle, N-S zenith angle, E-W amplitude, E-W phase, N-S amplitude, N-S phase, E-W zentih-angle window (from BSppm, Btest2. This measures the noisiness of the scattering point), and N-S zenith-angle window. The rest of my programs (like for winds) don't work until you've put these SPP strings through a program (discussed later) that will gather them into, say, 30min intervals, then reorder them by altitude.

13E. "Copy gafix.txt and bsppm.mbs to D:\final." I kept a copy of each gafix.txt (list of sounding times) and the SPPs in BSppM.mbs (renamed to, say, GR245.txt and GR245.mbs) on my hard drive, as well as backing them up to both DAT and EO later on. Be sure you rename the files so that you have them named by the AIDA tape number.

13F. "Copy gafix.txt, \Buffer02 files, and BSppm.mbs to DAT." I backed up the SPPs, the file names, and the calibrated tap (which is in \Buffer02 at this point) to DAT. Bookkeeping is miserable; needs a better system than mine. Sure is handy and fast to recover from DAT though.

13G. "Run SppMoxl on bsppm.mbs." The most useful tape-at-a-time diagnostic I have found is to plot the radial velocity vs altitude for all the scattering points, separately for the ordinary, extraordinary, and linear modes. SppMoxl.for will separate the tape-files SPPs into O, X, and L for plotting.

14. "Turn off screen copy." The screen output to this point is fairly condensed and makes a good diagnostic; you should survey it carefully to ensure that the entire processing went sensibly (you got no error messages during FltrB; you got scattering points where you expected them in BSppM, etc.)

15. "Print gafix.txt." I keep the list of files (which are the sounding times) as part of my hard-copy documentation on each tape; the plots of O, X, and L are the rest of it.

16. "Edit Sppm21o.grf, Sppm21X.grf, and Sppm21L.grf.' I had to enter the tape number and the times onto each graph (3 per tape) manually by editing the Golden-Graphics .grf file. It's pretty fast, but I'm sure there are better ways.

17. "Run PlotOXL.bat to generate 3 "21" plots." A 21 plot is SPP #2 (radial velocity) on the abscissa and #1 (altitude) on the ordinate. These three plots and the list of files are the hard-copy documentation I keep on each tape. (See Brosnahan and Adams, 1992, for samples of these polarization-filtered plots.)

18. "Update DatLog.txt." Since the DAT tapes require external bookkeeping, this is where I do it, on a one-page list (attached) called DatLog.txt. There's got to be a better way than this.

19. "Reboot with propper 10-tape disk in EO drive." I store 10 calibrated tapes and the associated SPP files on EO disks. Overkill.

20. "Copy \Buffer02, gafix.txt, and Bsppm.mbs to EO." But I do it anyhow.

Now the SPPs are in a format that is sounding time (10 4-byte words) followed by n SPPs (n lines of 10 4-byte numbers). I have a program called SGroup.for (subroutines SName.for, SMerge.for, and BellSub.for, which rings the bell a few times when the program is finished--it can be a slow program) that will read through a series of GRxxx.mbs files, group them into user-specified intervals (e.g. 10min, 2hour, etc.), and reorder them by altitude. This makes it a lot faster for wind calculations, but if I were doing it over, I'd let the wind program make multiple passes instead. This program is too complicated, and I've had trouble with it lately (the SPPs would come out close to, but not always exactly, ordered by altitude. Usually the error would be a fraction of a km; sometimes I'd find a 40km point up around 90km.) The program is designed to take a variety of inputs, but it's really hacked together. Sorry.

There are two versions of the wind-calculation program available. These are WindErr.for (subroutines SppFltr.for, Header.for, WFV.for, and WFH.for) and Wind.for (subroutines inName.for, outName.for, Header.for, WFV.for, WFH.for, and PhFit.for). (WFV = Wind-Fit, Vertical; WFH = Wind-Fit, Horizontal.) The first will do the 129 repeats of the calculations, with each variable (radial velocity, altitude, horizontal location) taking on its extreme values. This is done to determine the error bars due to calculational uncertainty. The second program uses just the nominal values for the input parameters to calculate the wind profile, but also calculates the components (sort of) of the velocity variance vector. This was just getting developed, so it will take some work. Probably best to junk my calculation and do it right. I don't do an actual fit to the perpendicular and parallel components of the velocity variance vector, but count them only if they're within 90 degrees (binary sorting). Works okay, but needs to be a full fit (which looks easy) before you do anything with the velocity variance vectors (which I'm sure carry all the information there is about the breaking waves).

```
copyin c:\buffer01\1.mbr/b:16384
copyin c:\buffer01\2.mbr/b:16384
copyin c:\buffer01\3.mbr/b:16384
copyin c:\buffer01\4.mbr/b:16384
copyin c:\buffer01\5.mbr/b:16384
copyin c:\buffer01\6.mbr/b:16384
copyin c:\buffer01\7.mbr/b:16384
copyin c:\buffer01\8.mbr/b:16384
copyin c:\buffer01\9.mbr/b:16384
copyin c:\buffer01\10.mbr/b:16384

copyin c:\buffer01\11.mbr/b:16384
copyin c:\buffer01\12.mbr/b:16384
copyin c:\buffer01\13.mbr/b:16384
copyin c:\buffer01\14.mbr/b:16384
copyin c:\buffer01\15.mbr/b:16384
copyin c:\buffer01\16.mbr/b:16384
copyin c:\buffer01\17.mbr/b:16384
copyin c:\buffer01\18.mbr/b:16384
copyin c:\buffer01\19.mbr/b:16384
copyin c:\buffer01\20.mbr/b:16384

copyin c:\buffer01\21.mbr/b:16384
copyin c:\buffer01\22.mbr/b:16384
copyin c:\buffer01\23.mbr/b:16384
copyin c:\buffer01\24.mbr/b:16384
copyin c:\buffer01\25.mbr/b:16384
copyin c:\buffer01\26.mbr/b:16384
copyin c:\buffer01\27.mbr/b:16384
copyin c:\buffer01\28.mbr/b:16384
copyin c:\buffer01\29.mbr/b:16384
copyin c:\buffer01\30.mbr/b:16384

copyin c:\buffer01\31.mbr/b:16384
copyin c:\buffer01\32.mbr/b:16384
copyin c:\buffer01\33.mbr/b:16384
copyin c:\buffer01\34.mbr/b:16384
copyin c:\buffer01\35.mbr/b:16384
copyin c:\buffer01\36.mbr/b:16384
copyin c:\buffer01\37.mbr/b:16384
copyin c:\buffer01\38.mbr/b:16384
copyin c:\buffer01\39.mbr/b:16384
copyin c:\buffer01\40.mbr/b:16384

copyin c:\buffer01\41.mbr/b:16384
copyin c:\buffer01\42.mbr/b:16384
copyin c:\buffer01\43.mbr/b:16384
copyin c:\buffer01\44.mbr/b:16384
copyin c:\buffer01\45.mbr/b:16384
copyin c:\buffer01\46.mbr/b:16384
copyin c:\buffer01\47.mbr/b:16384
copyin c:\buffer01\48.mbr/b:16384
copyin c:\buffer01\49.mbr/b:16384
copyin c:\buffer01\50.mbr/b:16384
rew/unl
```

# MOJO.BAT

This copies the calibrated SPP data to the DAT

```
:1
cd \Buffer01
tpu dt dfs tfs bl16384 dn"*.*"
cd \work

:2
: FixTape runs GAFix on the .mbr files in Buffer01, and puts
: the fixed files into Buffer02.  Data file names are in
: gafix.txt.
Call FixTape

:3
: FltrB does the dc removal and phase correction for the list of
: files in FltrB.txt. The filtered files are written to Buffer02.
del \Buffer01\*.mbr
copy gafix.txt FltrB.txt
FltrB

:4
: BsppM runs the SPP program on the list of files in BsppM.txt,
: which is the same as gafix.txt.
copy gafix.txt BsppM.txt
BsppM

:5
copy gafix.txt d:\final\gr215.txt
copy bsppm.mbs d:\final\gr215.mbs

:6
tpu dt dfs tfs bl16384 dn"gafix.txt"
cd \Buffer02
tpu dt dfs tfs bl16384 dn"*.*"
cd \work
tpu dt dfs tfs bl16384 dn"bsppm.mbs"

:7
sppmoxl bsppm.mbs

:8
sppmchk bsppm.mbs

Bell

time
```

```
del gafix.txt
echo off

:2
copy \buffer01\2.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:3
copy \buffer01\3.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:4
copy \buffer01\4.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:5
copy \buffer01\5.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:6
copy \buffer01\6.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:7
copy \buffer01\7.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:8
copy \buffer01\8.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:9
copy \buffer01\9.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:10
copy \buffer01\10.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:11
copy \buffer01\11.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:12
copy \buffer01\12.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:13
```

```
copy \buffer01\13.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:14
copy \buffer01\14.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:15
copy \buffer01\15.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:16
copy \buffer01\16.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:17
copy \buffer01\17.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:18
copy \buffer01\18.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:19
copy \buffer01\19.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:20
copy \buffer01\20.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:21
copy \buffer01\21.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:22
copy \buffer01\22.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:23
copy \buffer01\23.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:24
copy \buffer01\24.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:25
```

```
copy \buffer01\25.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:26
copy \buffer01\26.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:27
copy \buffer01\27.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:28
copy \buffer01\28.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:29
copy \buffer01\29.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:30
copy \buffer01\30.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:31
copy \buffer01\31.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:32
copy \buffer01\32.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:33
copy \buffer01\33.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:34
copy \buffer01\34.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:35
copy \buffer01\35.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:36
copy \buffer01\36.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:37
```

```
copy \buffer01\37.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:38
copy \buffer01\38.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:39
copy \buffer01\39.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:40
copy \buffer01\40.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:41
copy \buffer01\41.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:42
copy \buffer01\42.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:43
copy \buffer01\43.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:44
copy \buffer01\44.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:45
copy \buffer01\45.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:46
copy \buffer01\46.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:47
copy \buffer01\47.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:48
copy \buffer01\48.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:49
```

```
copy \buffer01\49.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

:50
copy \buffer01\50.mbr e:x.mbr
gafix e:x.mbr
del e:x.mbr

*******
goto 51
*******

:51
```

```
c
$Debug
c
      program GAFix
c
*********************************************************************
*                                                                   *
*    IDI Radar Utility Program                                      *
*    Copyright 1990, Holodyne Limited 1986                          *
*    All Rights Reserved                                            *
*                        March 2, 1991                              *
*                                                                   *
*********************************************************************
c
c
c  This program takes a data file from a MAPSTAR tape and creates
c  two files: a binary data files and an ascii header file.
c  Some information for the header file, such as the "TapeLabel",
c  must be user-input.
c  The data file inverts the byte order for PCs, if selected.
c  Data files are named by the date and time of each file:
c  YYMMHHMM.SSt.  Each header file is written to a YYHHMM.SSh
c  file and also to "Header.Dat".  The raw-data input file is specified
c  by the user on the command line.  "Pathout" is a character*12
c  variable that specifies the DOS path to the time-named output
c  files.  Header.dat is written to the default drive.
c
c  GAFix calls RdHdr, Names, and WrHdr
$Include:'GAFix.inc'
$Include:'Header.inc'
c
c  Invert is a flag for byte inversion.  Set Invert=0 for
c  sensible computers; -1 for PCs.
c
      icount = 0
      Invert = 1
      pathout = 'c:\buffer02\'
      open (1,file=' ',status='old',form='binary')
c
c  Read the header info
c
      read (1) (hdr(i),i=1,512)
c
c  FixHdr will extract from hdr all the radar-operating parameters.
c
      Call RdHdr
      icount = icount + 1
c     write (*,90001) icount,year,month,day,hour,minute,second
90001 format (1x,i2,'   TIME:                     ',6(i2,2x))
c
c  Names will generate the names of the output files from the date
c  and time info in the header.
c
      Call Names
c     write (*,90002) year,month,day,hour,minute,second
90002 format (' CORRECTED and CENTERED TIME:',6(i2,2x))

      write (*,90003) datafile
90003 format (' FILE BEING PREPARED: ',a24)
      open (2,file='gafix.txt')
```

```
20101 read (2,end=20102,fmt=90004) oldname
      go to 20101
90004 format (a24)

20102 backspace (2)
      write (2,90004) datafile
      close (2)
c
c  Write the Header file.
c
c       open (3,file=hdrfile,status='unknown')
        open (3,file='GAFix.hdr',status='unknown')
        Call WrHdr
        close (3)

c
c  Read the data.
c
      open (3,file=datafile,status='unknown',form='binary')
      PCount = 0
c     write (*,*) ' '
c     write (*,*) 'PROCESSING BLOCK # '
      do 10002 iblock=1,52
c     if ((iblock/13)*13 .ne. iblock) then
c     write (*,90201) iblock
c     else
c     write (*,90202) iblock
c     endif
90201 format (1x,i2,\)
90202 format (1x,i2)
      read (1) ((data(pulse,byte),
     1                byte=1,4),pulse=1,3968)
      PCount = PCount + 3968
c
c  If Invert=1, invert the byte order.
c
      if (Invert .eq. 1) then
      do 10001 pulse=1,3968
      hold = data(pulse,1)
      data(pulse,1) = data(pulse,4)
      data(pulse,4) = hold
      hold = data(pulse,2)
      data(pulse,2) = data(pulse,3)
      data(pulse,3) = hold
10001 continue
      endif
c
c  Write the data.
c
      if (PCount .lt. 206336) then
      write (3) ((data(pulse,byte),
     1                 byte=1,4),pulse=1,3968)
      read(1) (hdr(i),i=1,512)
      else
      write (3) ((data(pulse,byte),
     1                 byte=1,4),pulse=1,2432)
      go to 20002
      endif
10002 continue
20002 close (1)
```

```
close (2)
close (3)
close (4)
end
```

```
c
$Debug
c
      Subroutine RdHdr
c
******************************************************************
*                                                                *
*    IDI Radar Utility Program                                   *
*    Copyright 1991, Holodyne Limited 1986                       *
*    All Rights Reserved                                         *
*                       March 1, 1991                            *
*                                                                *
******************************************************************
c
c
c  This subroutine takes a 512-integer header string and reads
c  from it the radar operating parameters.  All parameters are
c  converted to MKS on input.
c
c  Link: GAFix RdHdr Names WrHdr
$Include:'GAFix.inc'
$Include:'Header.inc'

      do 10001 i=1,512
      hdr(i) = hdr(i) - 48
10001 continue
c
c  First Line of Header:
c
      SoundingNumber = hdr(3)*1000 + hdr(4)*100 + hdr(5)*10 + hdr(6)
      Year = hdr(14)*10 + hdr(15)
      Month = hdr(16)*10 + hdr(17)
      Day = hdr(18)*10 + hdr(19)
      Hour = hdr(21)*10 + hdr(22)
      Minute = hdr(23)*10 + hdr(24)
      Second = hdr(25)*10 + hdr(26)
c      write (SiteName,90001) hdr(i),i=33,46)
c90001 format (i1)
c      write (*,'(a14)') SiteName
      SiteName = 'Islote, P.R.  '
      DataType = '0001'
      DataMode = '0003'
      FFTPts = hdr(61)*1000 + hdr(62)*100
     1       + hdr(63)*10   + hdr(64)
c
c  Second Line of Header:
c
      Freq1 = hdr(78)*1e8 + hdr(79)*1e7 + hdr(80)*1e6
     1      + hdr(81)*1e5 + hdr(82)*1e4 + hdr(83)*1e3
     2      + hdr(84)*1e2 + hdr(85)*1e1 + hdr(86)
      Freq2 = hdr(91)*1e8 + hdr(92)*1e7 + hdr(93)*1e6
     1      + hdr(94)*1e5 + hdr(95)*1e4 + hdr(96)*1e3
     2      + hdr(97)*1e2 + hdr(98)*1e1 + hdr(99)
      PulseDuration = hdr(104)*1e-4 + hdr(105)*1e-5 + hdr(106)*1e-6
     1              + hdr(107)*1e-7 + hdr(108)*1e-8 + hdr(109)*1e-9
      FFTPeriod =  hdr(115)*1e+2 + hdr(116)*1e+1 + hdr(117)
     1           + hdr(118)*1e-1 + hdr(119)*1e-2 + hdr(120)*1e-3
      NumCohAve = hdr(125)*1e5 + hdr(126)*1e4 + hdr(127)*1e3
     1          + hdr(128)*1e2 + hdr(129)*1e1 + hdr(130)
c
```

```
c  Third Line of Header:
c
      NumRangeGates = hdr(136)*1000 + hdr(137)*100
     1               + hdr(138)*10   + hdr(139)
      NumRx = hdr(144)*10 + hdr(145)
      PRP = hdr(151)*1e-2 + hdr(152)*1e-3 + hdr(153)*1e-4
     1    + hdr(154)*1e-5 + hdr(155)*1e-6 + hdr(156)*1e-7
     2    + hdr(157)*1e-8 + hdr(158)*1e-9
c
c  Offset is the offset to the bottom of the first range-gate.
c  This is set by the radar operator, picked up automatically
c  by the system, and written to the header.  Offset in seconds:
c
      Offset = hdr(166)*1e-4 + hdr(167)*1e-5 + hdr(168)*1e-6
     1       + hdr(169)*1e-7 + hdr(170)*1e-8 + hdr(171)*1e-9
c
c  Delay is the equipment delay.  This depends on pulse length,
c  cable length, etc.  In Islote it was 7.4km.  This can
c  be entered by the operator into the radar's header, but that
c  didn't happen in Islote.
c
c      Delay = hdr(173)*1e-4 + hdr(174)*1e-5 + hdr(175)*1e-6
c     1      + hdr(176)*1e-7 + hdr(177)*1e-8 + hdr(178)*1e-9
c
      Delay =  7.4e3
      AltMin = Offset*1.5e8 - Delay
c
c  SS is the sample (range-gate) spacing.  This
c  is set by the radar operator, picked up automatically by the
c  system, and written to the header.
c
      SS = hdr(183)*1e-4 + hdr(184)*1e-5 + hdr(185)*1e-6
     1   + hdr(186)*1e-7 + hdr(187)*1e-8 + hdr(188)*1e-9
      AltStep = SS*1.5e8
      RxAttn = 0
      RxPolarization = 'L1'
c
c  Fourth Line of Header
c
      TxPower = 100000
      do 10002 i=1,10
      RxMask(i) = 1
10002 continue
      TapeLabel = 'SF030'
      return
      end
```

```
c
$Debug
c
      Subroutine Names
c
*******************************************************************
*                                                                 *
*    IDI Radar Utility Program                                    *
*    Copyright 1990, Holodyne Limited 1986                        *
*    All Rights Reserved                                          *
*                         March 1, 1991                           *
*                                                                 *
*******************************************************************
c
c
c  This subroutine creates the names for data and header files
c  from the header information.  Files are named by time at
c  center of the sounding, with clock correction.
c  Link: GAFix RdHdr Names WrHdr
$Include:'GAFix.inc'
$Include:'Header.inc'
      dimension DaysPerMo(12),DaysInMonth(12)
      character*2 ascmonth,ascday,aschour,ascminute,ascsecond
      real*4 Timecorrection
      real*8 BigTime
      integer*4 DaysPerMonth,DaysInMonth,spermin,sperhr,sperday
      character*1 char1,char2,char3
      data DaysPerMonth /31,28,31,30, 31, 30, 31, 31, 30, 31, 30, 31/
      data DaysInMonths  /0,31,59,90,120,151,181,212,243,273,304,334/

      char1 = '.'
      char2 = 't'
      char3 = 'h'

      spermin = 60
      sperhr = 3600
      sperday = 86400

      TimeCorrection = +40

      BigTime = Second + Minute*spermin + Hour*sperhr
     1        + (Day+DaysInMonths(month))*sperday
     1        + FFTPeriod/2 + TimeCorrection

      do 10002 i=1,11
      If (BigTime .lt. DaysInMonths(i+1)*SperDay) go to 20001
10002 continue
      write (*,*) 'Looks like a New Year to Me!'
      return

20001 Month = i
      BigTime = BigTime - sperday*DaysInMonths(month)

      Day = int(BigTime/sperday)
      BigTime = BigTime - Day*sperday
      Hour = int(BigTime/sperhr)
      BigTime = BigTime - Hour*sperhr
      Minute = int(BigTime/spermin)
      second = BigTime - Minute*spermin
c
```

```
c  Correct for temporal wrap-around.
c
20002 if (second .ge. 60) then
      Second = Second - 60
      Minute = Minute + 1
      go to 20002
      endif

20003 if (Minute .ge. 60) then
      Minute = Minute - 60
      Hour = Hour + 1
      go to 20003
      endif

20004 if (Hour .ge. 24) then
      Hour = Hour - 24
      Day = Day + 1
      go to 20004
      endif

20005 if (Day .gt. DaysPerMonth(Month)) then
      Day = Day - DaysPerMonth(Month)
      Month = Month + 1
      go to 20005
      endif

      if (month .lt. 10) then
      write (ascmonth,90001) '0',month
90001 format (a1,i1)
      else
      write (ascmonth,90002) month
90002 format (i2)
      endif

      if (day .lt. 10) then
      write (ascday,90001) '0',day
      else
      write (ascday,90002) day
      endif

      if (hour .lt. 10) then
      write (aschour,90001) '0',hour
      else
      write (aschour,90002) hour
      endif

      if (minute .lt. 10) then
      write (ascminute,90001) '0',minute
      else
      write (ascminute,90002) minute
      endif

      if (second .lt. 10) then
      write (ascsecond,90001) '0',second
      else
      write (ascsecond,90002) second
      endif

      write (datafile,90003)
     1 pathout,ascMonth,ascDay,ascHour,ascMinute,
```

```
     2 char1,ascsecond,char2

      write (hdrfile,90003)
     1 pathout,ascMonth,ascDay,ascHour,ascMinute,
     2 char1,ascsecond,char3

90003 format (24a)
      return
      end
```

```
c
$Debug
c
      Subroutine WrHdr
c
*******************************************************************
*                                                                 *
*    IDI Radar Utility Program                                    *
*    Copyright 1990, Holodyne Limited 1986                        *
*    All Rights Reserved                                          *
*                        May 5, 1990                              *
*                                                                 *
*******************************************************************
c
c  This subroutine writes the header file.
c  Link: GAFix RdHdr Names WrHdr
$Include:'GAFix.inc'
$Include:'Header.inc'

      write (3,90001) SoundingNumber
90001 format (1x,i4,20x,'SoundingNumber')

      write (3,90002) Year
90002 format (1x,i2,22x,'Year')

      write (3,90003) Month
90003 format (1x,i2,22x,'Month')

      write (3,90004) Day
90004 format (1x,i2,22x,'Day')

      write (3,90005) Hour
90005 format (1x,i2,22x,'Hour')

      write (3,90006) Minute
90006 format (1x,i2,22x,'Minute')

      write (3,90007) Second
90007 format (1x,i2,22x,'Second')

      write (3,90008) SiteName
90008 format (1x,a14,10x,'SiteName')

      write (3,90009) DataType
90009 format (1x,a4,20x,'DataType')

      write (3,90010) DataMode
90010 format (1x,a4,20x,'DataMode')

      write (3,90011) FFTPts
90011 format (1x,i4,20x,'FFTPts')

      write (3,90012) Freql
90012 format (1x,1Pe12.5,12x,'Freql (Hz)')

      write (3,90013) Freq2
90013 format (1x,1Pe12.5,12x,'Freq2 (Hz)')

      write (3,90014) PulseDuration
90014 format (1x,1Pe12.5,12x,'PulseDuration (seconds)')
```

```
      write (3,90015) FFTPeriod
90015 format (1x,1Pe12.5,12x,'FFTPeriod (seconds)')

      write (3,90016) NumCohAve
90016 format (1x,1Pe12.5,12x,'NumCohAve')

      write (3,90017) NumRangeGates
90017 format (1x,i4,20x,'NumRangeGates')

      write (3,90018) NumRx
90018 format (1x,i4,20x,'NumRx')

      write (3,90019) PRP
90019 format (1x,1Pe12.5,12x,'PRP (seconds)')

      write (3,90020) AltMin
90020 format (1x,1Pe12.5,12x,'AltMin (meters)')

      write (3,90021) AltStep
90021 format (1x,1Pe12.5,12x,'AltStep (meters)')

      write (3,90022) RxAttn
90022 format (1x,i4,20x,'RxAttn')

      write (3,90023) RxPolarization
90023 format (1x,a2,22x,'RxPolarization')

      write (3,90024) TxPower
90024 format (1x,1Pe12.5,12x,'TxPower (Watts)')

      write (3,90025) (RxMask(i),i=1,NumRx)
90025 format (10(1x,i1),5x,'RxMask')

      write (3,90026) TapeLabel
90026 format (1x,a5,19x,'TapeLabel')

      close (2)
      return
      end
```

GAFix.inc

```
c
c  GAFix.inc
c
      common /GAFix1/ hdr(512),data(3968,4)
      common /GAFix2/ hold,invert,range,pathout
      integer*1 hdr,data,hold
      integer*2 invert,range,pulse,byte
      integer*4 PCount
      character*12 pathout
```

Header.inc

```
c
c  Header.inc
c
      common /H/ SoundingNumber,Year,Month,Day,Hour,Minute,Second,
     1                SiteName,DataType,DataMode,FFTpts,Freq1,Freq2,
     2                PulseDuration,FFTPeriod,NumCohAve,NumRangeGates,
     3                NumRx,PRP,AltMin,AltStep,RxAttn,RxPolarization,
     4                TxPower,RxMask(10),datafile,TapeLabel
      integer*4 SoundingNumber,Year,Month,Day,Hour,Minute,Second
      character*14 SiteName
      character*4 DataType,DataMode
      integer*4 FFTpts
      real*4 Freq1,Freq2,PulseDuration,FFTPeriod,NumCohAve
      integer*4 NumRangeGates,NumRx
      real*4 PRP,AltMin,AltStep
      integer*4 RxAttn
      character*2 RxPolarization
      real*4 TxPower
      integer*4 RxMask
      character*24 datafile,hdrfile,snake
      character*5 TapeLabel
```

```
c
$DEBUG
c
      program FltrB
C
C   FltrB ( = Filter B) is a replacement for Fltr to use when you
c   can't apply the pulse-256 correction, i.e., when you've lost
c   track, due to missing files or whatever, of the pulse-256 offset.
c   FltrB is a 4-pass process that will (1) remove the local DC
c   offsets for each file, range, receiver, and quadrature channel,
c   (2) remove single-pulse noise spikes, (3) calculate the receiver
c   receiver gains, and (4) adjust the gains and phases of the receivers.
c   A list of input files are expected in FltrB.txt.
c
C   Link FltrB+Header+FltrB1+FltrB2+FltrB3+FltrB4
C
$Include:'Header.inc'
$Include:'FltrB.inc'
      Dimension FileA(60),FileB(60)
      CHARACTER*24 FileA,FileB


c
c   get the names of the input and output files.
c
      open (1,file='FltrB.txt',status='old')
      ifile = 1
20001 read (1,end=20002,fmt=90001) FileA(ifile)
90001 format (a24)
      ifile = ifile + 1
      go to 20001
20002 Numfiles = ifile - 1
      datafile = fileA(1)
      call Header
      write (*,*) 'NumFiles = ',NumFiles
      close (1)
      do 10001 ifile = 1,NumFiles
      Snake = FileA(ifile)
      write (FileB(ifile),90002) 'c:\Buffer01\',Snake(13:24)
90002 format (a12,a12)
      write (*,*) ifile,'  ',FileA(ifile),'  ',FileB(ifile)
10001 continue

*************************************************************
c
c Filter #1.  Remove local DC offsets.
c
      do 10011 ifile = 1,NumFiles
      infile(ifile) = FileA(ifile)
      outfile(ifile) = FileB(ifile)
10011 continue
      write (*,*) 'FltrB1:'
      Call FltrB1

*************************************************************
c
c Filter #2.  Remove noise spikes.
c
      do 10021 ifile = 1,NumFiles
      infile(ifile) = FileB(ifile)
      outfile(ifile) = FileA(ifile)
```

```
10021 continue

      write (*,*) 'FltrB2:'
      Call FltrB2

*************************************************************
c
c  Filter #3.  Calculate receiver gains.
c
      do 10031 ifile = 1,NumFiles
      infile(ifile) = FileA(ifile)
      outfile(ifile) = FileB(ifile)
10031 continue

      write (*,*) 'FltrB3:'
      Call FltrB3

*************************************************************
c
c  Filter #4.  Adjust receiver gains and phases.
c
      do 10041 ifile = 1,NumFiles
      infile(ifile) = FileB(ifile)
      outfile(ifile) = FileA(ifile)
10041 continue

      write (*,*) 'FltrB4:'
      Call FltrB4
**************************************************************

90909 end
```

```
c
$DEBUG
c
      Subroutine FltrB1
C
C   FltrB1( = Filter #B1) will identify and remove DC offsets.
c
C   Link FltrB+Header+FltrB1+FltrB2+FltrB3+FltrB4
C
$Include:'Header.inc'
$Include:'FltrB.inc'
      Dimension power(10,512),pdB(10,512),
     1          RunAve(40,10),dc1(10,2),dc2(10,2),
     2          NumAve(10),erms(10,2),NumNoise(10)
      Real*4 x1,x2,y1,y2,dc1,dc2,erms,errorx,errory,alpha,beta,sigma
      Integer*4 Finish,NumBig,pend,NumAve,NumNoise
c
c  Alpha is the inverse width of the exponential running average.
c  Sigma*erms is the quadrature error criterion for inclusion in
c  the dc average.
c
      Alpha = 0.1
      Sigma = 2
      do 10402 ifile = 1,Numfiles
      if (((ifile/25)*25 .eq. ifile) .or. (ifile .eq. Numfiles)) then
      write (*,90002) ifile
      else
      write (*,90001) ifile
      endif
90001 format (1x,i2,\)
90002 format (1x,i2)

      open (1,file=infile(ifile),status='old',form='binary')
      open (2,file=outfile(ifile),status='unknown',form='binary')
c     write (*,*) 'In FltrB1: ifile,outfile = ',ifile,outfile(ifile)
      do 10401 range = 1,NumRangeGates
c
c  3-pass dc average.  First Pass: get raw dc average (dc1).
c
      do 10101 rx = 1,NumRx
      NumAve(rx) = 0
      NumNoise(rx) = 0
      do 10101 quad = 1,2
      dc1(rx,quad) = 0
      dc2(rx,quad) = 0
      DCave(rx,quad) = 0
10101 continue
      read (1) (((data(rx,pulse,quad),quad=1,2),
     1                          pulse=1,FFTPts),
     2                              rx=1,NumRx)
      do 10102 rx = 1,NumRx
      do 10102 pulse = 1,FFTPts
      do 10102 quad = 1,2
      dc1(rx,quad) = dc1(rx,quad) + float(data(rx,pulse,quad))
10102 continue
      do 10103 rx = 1,NumRx
      do 10103 quad = 1,2
      dc1(rx,quad) = dc1(rx,quad)/FFTPts
10103 continue
c
```

```
c  Second pass: get rms deviation from average.
c
      do 10201 rx = 1,NumRx
      do 10201 pulse = 1,FFTPts
      do 10201 quad = 1,2
      erms(rx,quad) = erms(rx,quad)
     1    + (dc1(rx,quad)-float(data(rx,pulse,quad)))**2
10201 continue
      do 10202 rx = 1,NumRx
      do 10202 quad = 1,2
      erms(rx,quad) = sqrt(erms(rx,quad)/FFTPts)
10202 continue
c
c  Third pass: recalculate the dc average; exclude points that
c  lie more than sigma times the rms deviation.
c
      do 10301 rx = 1,NumRx
      do 10301 pulse = 1,FFTPts
      errorx = abs(dc1(rx,1)-float(data(rx,pulse,1)))
      errory = abs(dc1(rx,2)-float(data(rx,pulse,2)))
      if (errorx .lt. Sigma*erms(rx,1) .and.
     1    errory .lt. Sigma*erms(rx,2)) then
      dc2(rx,1) = dc2(rx,1) + float(data(rx,pulse,1))
      dc2(rx,2) = dc2(rx,2) + float(data(rx,pulse,2))
      NumAve(rx) = NumAve(rx) + 1
      endif
10301 continue
      do 10302 rx = 1,NumRx
      do 10302 quad = 1,2
      if (NumAve(rx) .gt. 0) then
      dc2(rx,quad) = dc2(rx,quad)/float(NumAve(rx))
      else
      write (*,*) 'rx,quad,NumAve = ',rx,quad,NumAve(rx)
      endif
10302 continue
      do 10303 rx = 1,NumRx
      NumAve(rx) = 0
      do 10303 pulse = 1,FFTPts
      do 10303 quad = 1,2
      data(rx,pulse,quad) = data(rx,pulse,quad) - nint(dc2(rx,quad))
10303 continue

      write (2) (((data(rx,pulse,quad),quad=1,2),
     1                             pulse=1,FFTPts),
     2                                 rx=1,NumRx)
10401 continue
      close (1)
      close (2)
10402 continue

      return
      end
```

```
c
$DEBUG
c
      Subroutine FltrB2
C
C   FltrB2( - Filter #B2) will identify and remove noise bursts.
c   By definition here,
c   an increase in signal power in one receiver is a noise burst if
c   its power exceed the criterion tor a single pulse. (Fix this!!!)
c   The noise-burst data are replaced with a linear
c   interpolation.
c   Receivers are treated individually for noise elimination.
c   FltrB2 uses an exponential running average to look for noise
c   bursts.  To get starting values for the running
c   average of each sounding, we use the arithmetic average.
c   Beta is the number of dB above the running average for the
c   point to be considered too big.
c
C   Link FltrB+Header+FltrB1+FltrB2+FltrB3+FltrB4
C
$Include:'Header.inc'
$Include:'FltrB.inc'
      Common power(10,512),pdB(10,512)
      Dimension RunAve(40,10),NumAve(10),NumNoise(10)
      Real*4 x1,x2,y1,y2,dc1,dc2,erms,error,alpha,beta,sigma
      Integer*4 Finish,NumBig,pend,NumAve,NumNoise
c
c  Alpha is the inverse width of the exponential running average.
c  Beta is the number of dB above the running average for a point to
c  be declared a noise burst, and removed from the data.
c
      Alpha - 0.1
      Beta - 10

      do 10409 ifile - 1,Numfiles
      if (((ifile/25)*25 .eq. ifile) .or. (ifile .eq. Numfiles)) then
      write (*,90002) ifile
      else
      write (*,90001) ifile
      endif
90001 format (1x,i2,\)
90002 format (1x,i2)

      open (1,file-infile(ifile),status-'old',form-'binary')
      if (ifile .lt. Numfiles)
     1  open (2,file-infilein(ifile+1),status-'old',form-'binary')
      open (3,file-outfile(ifile),status-'old',form-'binary')
c     write (*,*) 'FltrB2: ifile,infile,outfile - ',
c    1                        ifile,infile(ifile),outfile(ifile)

      do 10408 range - 1,NumRangeGates
*************************************************************
c
c                  Fill the Arrays
c
c  Fill the Left-Hand Side of the Pulse String from file #1
c  and the RHS from file #2.  On last file, skip #2.
c
      read (1) (((data(rx,pulse,quad),quad-1,2),
     1                           pulse-1,FFTPts),
```

```
     2                                      rx=1,NumRx)
      if (ifile .lt. Numfiles) then
      read (2) (((data(rx,pulse,quad),quad=1,2),
     1                 pulse=FFTPts+1,2*FFTPts),
     2                                      rx=1,NumRx)
      pend = 2*FFTPts
      else
      pend = FFTPts
      endif
      do 10401 rx = 1,NumRx
      do 10401 pulse = 1,pend
      power(rx,pulse) = float(data(rx,pulse,1))**2
     1                + float(data(rx,pulse,2))**2
      if (power(rx,pulse) .ge. 1) then
      pdB(rx,pulse) = 10*log10(power(rx,pulse))
      else
      pdB(rx,pulse) = 1
      endif
10401 continue

      if (ifile .eq. 1) then
      do 10403 rx = 1,NumRx
      RunAve(range,rx) = 0
      do 10402 pulse = 1,pend
      RunAve(range,rx) = RunAve(range,rx) + pdB(rx,pulse)
10402 continue
      RunAve(range,rx) = RunAve(range,rx)/pend
10403 continue
      endif

*************************************************************
c
c  Check the data point by point; look for noise.
c
      do 10407 rx = 1,NumRx
      W = Beta + RunAve(range,rx)
      if (ifile .eq. 1) then
      if (pdB(rx,1) .gt. W .and. pdB(rx,2) .le. W) then
      data(rx,1,1) = data(rx,2,1)
      data(rx,1,2) = data(rx,2,2)
      NumNoise(rx) = NumNoise(rx) + 1
      else
      RunAve(range,rx) = RunAve(range,rx)*(1-alpha)
     1                 + pdB(rx,1)*alpha
      endif
      endif

      if (ifile .lt. NumFiles) pend = FFTPts+1
      if (ifile .eq. NumFiles) pend = FFTPts-1
      do 10406 pulse = 2,pend
      if ((pdB(rx,pulse-1) .le. W) .and.
     1   (pdB(rx,pulse) .gt. W) .and.
     2   (pdB(rx,pulse+1) .le. W)) then
      x1 = float(data(rx,pulse-1,1))
      x2 = float(data(rx,pulse+1,1))
      y1 = float(data(rx,pulse-1,2))
      y2 = float(data(rx,pulse+1,2))
      data(rx,pulse,1) = nint((x1+x2)/2.)
      data(rx,pulse,2) = nint((y1+y2)/2.)
      NumNoise(rx) = NumNoise(rx) + 1
```

```
      else
      RunAve(range,rx) = RunAve(range,rx)*(1-alpha)
     1                 + pdB(rx,pulse)*alpha
      endif
10406 continue

      if (ifile .eq. NumFiles) then
      if (pdB(rx,FFTPts-1). lt. W .and. pdB(rx,FFTPts) .ge. W) then
      data(rx,FFTPts,1) = data(rx,FFTPts-1,1)
      data(rx,FFTPts,2) = data(rx,FFTPts-1,2)
      NumNoise(rx) = NumNoise(rx) + 1
      else
      RunAve(range,rx) = RunAve(range,rx)*(1-alpha)
     1                 + pdB(rx,FFTPts)*alpha
      endif
      endif

10407 continue
      write (3) (((data(rx,pulse,quad),quad=1,2),
     1                            pulse=1,FFTPts),
     2                                rx=1,NumRx)
10408 continue
      close (1)
      close (2)
10409 continue

      do 10412 rx = 1,NumRx
      write (*,*) rx,NumNoise(rx)
10412 continue
      close (3)
      return
      end
```

```
c
$DEBUG
c
      Subroutine FltrB3
c
c   FltrB3( = Filter #B3) will calculate the channel voltage gains.
c
c   Link FltrB+Header+FltrB1+FltrB2+FltrB3+FltrB4
c
$Include:'header.inc'
$Include:'FltrB.inc'
      Dimension power(10,2)
      Integer*4 RefRx,RefQuad,RGL,RGU
      RefRx = 5
      RefQuad = 1
      RGL = 31
      RGU = 40
      do 10003 ifile = 1,NumFiles
      if (((ifile/25)*25 .eq. ifile) .or. ifile .eq. Numfiles) then
      write (*,90002) ifile
      else
      write (*,90001) ifile
      endif
90001 format (1x,i2,\)
90002 format (1x,i2)

      open (1,file=infile(ifile),form='binary')
      open (2,file=outfile(ifile),form='binary')
      do 10002 range = 1,NumRangeGates
      read (1) (((data(rx,pulse,quad),quad=1,2),
     1                            pulse=1,FFTPts),
     2                                rx=1,NumRx)

      do 10001 rx = 1,NumRx
      do 10001 pulse = 1,FFTPts
      do 10001 quad = 1,2
      if (range .ge. RGL .and. range .le. RGU)
     1    power(rx,quad) = power(rx,quad)
     2                   + float(data(rx,pulse,quad))**2
10001 continue

      write (2) (((data(rx,pulse,quad),quad=1,2),
     1                            pulse=1,FFTPts),
     2                                rx=1,NumRx)

10002 continue
      close (1)
      close (2)
10003 continue

      do 10011 rx = 1,NumRx
      do 10011 quad = 1,2
      Vgain(rx,quad) = sqrt(power(rx,quad)/power(RefRx,RefQuad))
10011 continue
      write (*,*) 'Voltage Gains:'
      write (*,*) '        rx       quad      Vgain'
      do 10021 rx = 1,NumRx
      do 10021 quad = 1,2
      write (*,*) rx,quad,Vgain(rx,quad)
10021 continue
```

```
99999 end
```

```
c
$DEBUG
c
      Subroutine FltrB4
c
c   FltrB4( = Filter #B4) will adjust the receiver voltage gains and
c   phase offsets
c
c   Link FltrB+Header+FltrB1+FltrB2+FltrB3+FltrB4
c
$Include:'header.inc'
$Include:'FltrB.inc'
      Dimension Correct(10)
      complex*8 V,srmo
      Real*4 x,y,vtemp
      srmo = cmplx(0,1)
      pi = 3.1415927
      contor = pi/180

      Correct(1) = 0
      Correct(2) = 0
      Correct(3) = 0
      Correct(4) = 0
      Correct(5) = 0
      Correct(6) = 0
      Correct(7) = 0
      Correct(8) = 0
      Correct(9) = 0
      Correct(10) = 0
c
c  experimental set to adjust to airplane.
c
      Correct(1) = contor * (-17.86)
      Correct(2) = contor * (66.23 - 90 - 25)
      Correct(3) = contor * (41.16)
      Correct(4) = contor * (86.92 - 90 - 25)
      Correct(5) = contor * (0.00)
      Correct(6) = contor * (74.71 - 90 - 25)
      Correct(7) = contor * (-26.81 - 15)
      Correct(8) = contor * (71.20 - 90 - 25 -10)
      Correct(9) = contor * (-14.01)
      Correct(10) =contor * (75.71 - 90 - 25 + 10)

c
c  The following corrections from an average over 5 tapes
c  (Gr-237,238,239,242,254).
c  These judged best: 11/30/89.
c
      Correct(1) = contor * (-17.86)
      Correct(2) = contor * (66.23 - 90)
      Correct(3) = contor * (41.16)
      Correct(4) = contor * (86.92 - 90)
      Correct(5) = contor * (0.00)
      Correct(6) = contor * (74.71 - 90)
      Correct(7) = contor * (-26.81)
      Correct(8) = contor * (71.20 - 90)
      Correct(9) = contor * (-14.01)
      Correct(10) =contor * (75.71 - 90)

      do 10006 ifile = 1,NumFiles
```

```
      if (((ifile/25)*25 .eq. ifile) .or. ifile .eq. Numfiles) then
      write (*,90002) ifile
      else
      write (*,90001) ifile
      endif
90001 format (1x,i2,\)
90002 format (1x,i2)

      open (1,file=infile(ifile),form='binary')
      open (2,file=outfile(ifile),form='binary')
      do 10005 range = 1,NumRangeGates
      read (1) (((data(rx,pulse,quad),quad=1,2),
     1                           pulse=1,FFTPts),
     2                               rx=1,NumRx)

      do 10002 rx = 1,NumRx
      do 10002 pulse = 1,FFTPts
c
c  Correct voltage gains.
c
      do 10001 quad = 1,2
      vtemp = float(data(rx,pulse,quad))
      data(rx,pulse,quad) = nint(vtemp/Vgain(rx,quad))
10001 continue
c
c  Correct phases.
c
      x = float(data(rx,pulse,1))
      y = float(data(rx,pulse,2))
      V = cmplx(x,y)*cexp(+srmo*correct(rx))
      data(rx,pulse,1) = nint(real(V))
      data(rx,pulse,2) = nint(aimag(V))
10002 continue

      write (2) (((data(rx,pulse,quad),quad=1,2),
     1                           pulse=1,FFTPts),
     2                               rx=1,NumRx)
10005 continue
      close (1)
      close (2)
10006 continue
99999 end
```

```
c
$Debug
c
      Program BSPPM
c
*******************************************************************
*                                                                 *
*     Scattering-Point Parameter Analysis Program                 *
*     Copyright 1989, Holodyne Limited 1986.                      *
*     All Rights Reserved.                                        *
*                                          December 8, 1989       *
*******************************************************************
c
c   Format for Scattering-Point Parameters:
c   1.  Altitude (km).
c   2.  Radial velocity (m/sec).
c   3.  Zenith angle in East-West meridian (degrees).
c   4.  Zenith angle in North-South meridian (degrees).
c   5.  Voltage amplitude on Dipole #1 (East-pointing);
c       sum of 5 steered voltages.
c   6.  Phase on Dipole #1 at vertex of array (degrees).
c   7.  Voltage amplitude on Dipole #2 (North-pointing);
c       sum of 5 steered voltages.
c   8.  Phase on Dipole #1 at vertex of array (degrees).
c   9.  Width of E-W zenith-angle window.
c   10. Width of N-S zenith-angle window.
c
c  bsppM.for calls bfftM,fft2cm,header,btestM,bsteerM, and bsortM.
c
c  March 10, 1991
c  Scattering-point parameters 9 & 10 added: 3/18/91.
c
$Include:'Bsppm.inc'
$Include:'header.inc'
      real*4 ZAWdegrees
      integer*4 SPPbyRange,NoiseCount,NoiseLimit
      srmo = cmplx(0,1)
      pi = 3.14159265
      Clight = 3e8
      contod = 180/pi
      contor = 1/contod
      AntLocation(1) = -1.0
      AntLocation(2) = -0.5
      AntLocation(3) =  0.0
      ZAWdegrees = 20
      ZAWindow = ZAWdegrees*contor
      Threshold = 1e-2
      LPswitch = 0
      TxPolarization = 'L'
      NoiseLimit = 255*5

      open (3,file='Bsppm.mbs',form='binary')
      open (1,file='Bsppm.txt')
20001 read (1,fmt=90001,end=90909) datafile
      write (*,*) ' '
      write (*,90002) datafile
90001 format (a24)
90002 format (' Processing',1x,a24)
c     write (*,90003) ZAWdegrees,Threshold,NoiseLimit
90003 format (1x,'ZAWindow = ',f4.1,'       Threshold = ',1Pe6.0,
```

```
      1              '       NoiseLimit = ',i4)
       Call Header
       DataMode = '0003'
       open (2,file=datafile,status='old',form='binary')
       SPPnumber = 0
       SumPower = 0
       ibelly = 0
       NoiseCount = 0

       do 10001 ireject = 1,12
       reject(ireject) = 0
 10001 continue

       write (*,*) 'Number of Scattering Points:'
       Do 10004 jRange = 1,NumRangeGates
       SPPbyRange = 0

       read (2) ((((Data(1,ant,dipole,pulse,quad), quad=1,2),
      1                                          pulse=1,256),
      2                                          dipole=1,2),
      3                                          ant=1,3)
       read (2) (((Data(2,2,dipole,pulse,quad), quad=1,2),
      1                                       pulse=1,256),
      2                                       dipole=1,2)
       read (2) (((Data(2,1,dipole,pulse,quad), quad=1,2),
      1                                       pulse=1,256),
      2                                       dipole=1,2)

       do 10002 quad = 1,2
       do 10002 pulse = 1,256
 c  (Channels 3 and 4 are reversed on tapes 44-88.)
 c     iswap = data(1,2,1,pulse,quad)
 c     data(1,2,1,pulse,quad) = data(1,2,2,pulse,quad)
 c     data(1,2,2,pulse,quad) = iswap
       do 10002 dipole = 1,2
       Data(2,3,dipole,pulse,quad) = Data(1,3,dipole,pulse,quad)
 10002 continue

       Call BFFTM

       do 10003 dopp = 1,256
       if (dopp .eq. 128) go to 10003
       failflag = 0
       Call BTESTM
       if (failflag .eq. 1) go to 10003
       Range = (AltMin + (jRange-1)*AltStep)*1e-3
       if (Range .le. 0) go to 10003
 c
 c  Real scattering point.
 c
       Altitude = Range*SqRt(1-Sin(ThetaEW)**2
      1                         -Sin(ThetaNS)**2)
       FDopp = (dopp-128.)/(FFTPeriod)
       VDopp = FDopp*CLight/(2*Freq1)
       Call BSTEERM
 c  Filters, if any, go here (if fail go to 10003):
 c
 c  End filter. Fill the next SPPtemp slot with the 10 SPPs.
       SPPnumber = SPPnumber + 1
       SPPbyRange = SPPbyRange + 1
```

```
      SPPtemp(SPPnumber,1) = Altitude
      SPPtemp(SPPnumber,2) = VDopp
      if (DataMode .eq. '0002') then
      SPPtemp(SPPnumber,3) = ThetaEW*contod
      SPPtemp(SPPnumber,4) = ThetaNS*contod
      elseif (DataMode .eq. '0003') then
      rotate = pi/4
      sinEW = sin(ThetaEW)
      sinNS = sin(ThetaNS)
      sinZA = sqrt(sinEW**2 + sinNS**2)
      if ((abs(sinEW) .gt. le-5) .or. (abs(sinNS) .gt. le-5)) then
      phinew = atan2(sinNS,sinEW) + rotate
      else
      phinew = 0
      endif
      SPPtemp(SPPnumber,3) =
     1    contod*asin(sinZA*cos(phinew))
      SPPtemp(SPPnumber,4) =
     1  contod*asin(sinZA*sin(phinew))
      endif

      spptemp(SPPnumber,5) = VAmplitude(1)
      spptemp(SPPnumber,6) = Faze(1)*contod
      spptemp(SPPnumber,7) = VAmplitude(2)
      spptemp(SPPnumber,8) = Faze(2)*contod
      spptemp(SPPnumber,9) = ZASpread(1)*contod/3
      spptemp(SPPnumber,10) = ZASpread(2)*contod/3
c  spptemp overflow protection:
      if (SPPnumber .eq. 2500) then
      write (*,*) 'Hit 2500 scattering points.  Full belly.'
      ibelly = 1
      go to 20002
      endif

10003 continue

20002 if ((jRange/10)*10 .ne. jRange) then
      write (*,90004) sppByRange
      else
      write (*,90005) SppByRange
      endif
      if (jRange .ge. 11 .and. jRange .le. 15)
     1    NoiseCount = NoiseCount + SppByRange
90004 format (1x,i5,\)
90005 format (1x,i5)
      if (ibelly .eq. 1) go to 20003
10004 continue
20003 close (2)
      write (*,*) 'Total # of scattering points found: ',sppnumber

      if ((NoiseCount .lt. NoiseLimit) .and. (sppnumber .gt. 0)) then
      S(1) = -999
      S(2) = Year
      S(3) = Month
      S(4) = Day
      S(5) = Hour
      S(6) = Minute
      S(7) = Second
      S(8) = sppnumber
      s(9) = -999
```

```
      s(10) = -999
      write (3) (S(is),is=1,10)

      Call BSORTM
      else
      write (*,*) 'Rejected.  NoiseCount = ',NoiseCount
      endif
c      write (*,*) 'Rejection Statistics:'
c      write (*,*)
c     1'    1     2     3     4     5     6     7     8     9    10    11'
c      write (*,90006) (reject(ireject),ireject=1,11)
90006 format (1x,11(i6))
      go to 20001
90909 close (1)
      close (2)
      close (3)
c     call BellSub
      end
```

```
c
$Debug
c
      Subroutine BFFTM
c
$include:'BsppM.inc'
$include:'Header.inc'

      dimension a(256),iwk(9)
      complex*16 a
      real*4 xend,yend
      integer*2 dp,dp2
      integer*4 iwk
      do 10004 dir=1,2
      do 10004 dipole = 1,2
      Voltage(dir,dipole) = 0
      do 10004 ant=1,3
c
c  replace the endpoints by the average endpoint.
c
      xend = float(data(dir,ant,dipole,1,1)+data(dir,ant,dipole,256,1))
      yend = float(data(dir,ant,dipole,1,2)+data(dir,ant,dipole,256,2))
      data(dir,ant,dipole,1,1) = nint(xend/2.0)
      data(dir,ant,dipole,256,1) = nint(xend/2.0)
      data(dir,ant,dipole,1,2) = nint(yend/2.0)
      data(dir,ant,dipole,256,2) = nint(yend/2.0)
c
c  transfer the data into "a" and perform the fft.
c
      do 10001 dp=1,256
      a(dp) = cmplx(data(dir,ant,dipole,dp,1),data(dir,ant,dipole,dp,2))
10001 continue

      call fft2cm(a,8,iwk)
c
c  find the total power.
c
      psum = 0
      do 10002 dp = 1,256
      x = float(Data(dir,ant,dipole,dp,1))
      y = float(Data(dir,ant,dipole,dp,2))
      psum = psum + x**2 + y**2
10002 continue

      do 10003 dp = 1,256
      p = cabs(a(dp))**2
      if (p .ge. psum*THRESHOLD) then
      xdata(dir,ant,dipole,dp,1) = real(a(dp))
      xdata(dir,ant,dipole,dp,2) = aimag(a(dp))
      else
      xdata(dir,ant,dipole,dp,1) = 0
      xdata(dir,ant,dipole,dp,2) = 0
      endif
10003 continue
10004 continue
      return
      end
```

```
C-----------------------------------------------------------------
C
C   COMPUTER             - PC
C
C   PURPOSE              - COMPUTE THE FAST FOURIER TRANSFORM OF A
C                            COMPLEX VALUED SEQUENCE OF LENGTH EQUAL TO
C                            A POWER TWO
C
C   USAGE                - CALL FFT2CM (A,M,IWK)
C
C   ARGUMENTS     A      - COMPLEX VECTOR OF LENGTH N, WHERE N=2**M.
C                            ON INPUT A CONTAINS THE COMPLEX VALUED
C                            SEQUENCE TO BE TRANSFORMED.
C                            ON OUTPUT A IS REPLACED BY THE
C                            FOURIER TRANSFORM.
C                 M      - INPUT EXPONENT TO WHICH 2 IS RAISED TO
C                            PRODUCE THE NUMBER OF DATA POINTS, N
C                            (I.E. N = 2**M).
C                 IWK    - WORK VECTOR OF LENGTH M+1.
C
C   REMARKS  1.   FFT2CM COMPUTES THE FOURIER TRANSFORM, X, ACCORDING
C                 TO THE FOLLOWING FORMULA;
C
C                   X(K+1) = SUM FROM J = 0 TO N-1 OF
C                            A(J+1)*CEXP((0.0,(2.0*PI*J*K)/N))
C                   FOR K=0,1,...,N-1 AND PI=3.1415...
C
C                 NOTE THAT X OVERWRITES A ON OUTPUT.
C            2.   FFT2CM CAN BE USED TO COMPUTE
C
C                   X(K+1) = (1/N)*SUM FROM J = 0 TO N-1 OF
C                            A(J+1)*CEXP((0.0,(-2.0*PI*J*K)/N))
C                   FOR K=0,1,...,N-1 AND PI=3.1415...
C
C                 BY PERFORMING THE FOLLOWING STEPS;
C
C                      DO 10 I=1,N
C                         A(I) = CONJG(A(I))
C                   10 CONTINUE
C                      CALL FFT2CM (A,M,IWK)
C                      DO 20 I=1,N
C                         A(I) = CONJG(A(I))/N
C                   20 CONTINUE
C
C-----------------------------------------------------------------
C
      SUBROUTINE  FFT2CM (A,M,IWK)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER*4           M
      INTEGER*4           IWK(1)
      COMPLEX*16          A(1)
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER*4           I,ISP,J,JJ,JSP,K,K0,K1,K2,K3,KB,KN,MK,MM,MP,N,
     1                    N4,N8,N2,LM,NN,JK
      DOUBLE PRECISION    RAD,C1,C2,C3,S1,S2,S3,CK,SK,SQ,A0,A1,A2,A3,
     1                    B0,B1,B2,B3,TWOPI,TEMP,
     2                    ZERO,ONE,Z0(2),Z1(2),Z2(2),Z3(2)
      COMPLEX*16          ZA0,ZA1,ZA2,ZA3,AK2,sort(16384)
```

```
      EQUIVALENCE         (ZA0,Z0(1)),(ZA1,Z1(1)),(ZA2,Z2(1)),
     1                    (ZA3,Z3(1)),(A0,Z0(1)),(B0,Z0(2)),(A1,Z1(1)),
     2                    (B1,Z1(2)),(A2,Z2(1)),(B2,Z2(2)),(A3,Z3(1)),
     3                    (B3,Z3(2))
      DATA                SQ/.70710678118654750D0/,
     1                    SK/.38268343236508980D0/,
     2                    CK/.92387953251128680D0/,
     3                    TWOPI/6.2831853071795860D0/
      DATA                ZERO/0.0D0/,ONE/1.0D0/
C                                   SQ=SQRT2/2,SK=SIN(PI/8),CK=COS(PI/8)
C                                   TWOPI=2*PI
C                                   FIRST EXECUTABLE STATEMENT
      MP = M+1
      N = 2**M
      IWK(1) = 1
      MM = (M/2)*2
      KN = N+1
C                                   INITIALIZE WORK VECTOR
      DO 5  I=2,MP
         IWK(I) = IWK(I-1)+IWK(I-1)
    5 CONTINUE
      RAD = TWOPI/N
      MK = M - 4
      KB = 1
      IF (MM .EQ. M) GO TO 15
      K2 = KN
      K0 = IWK(MM+1) + KB
   10 K2 = K2 - 1
      K0 = K0 - 1
      AK2 = A(K2)
      A(K2) = A(K0) - AK2
      A(K0) = A(K0) + AK2
      IF (K0 .GT. KB) GO TO 10
   15 C1 = ONE
      S1 = ZERO
      JJ = 0
      K = MM - 1
      J = 4
      IF (K .GE. 1) GO TO 30
      GO TO 70
   20 IF (IWK(J) .GT. JJ) GO TO 25
      JJ = JJ - IWK(J)
      J = J-1
      IF (IWK(J) .GT. JJ) GO TO 25
      JJ = JJ - IWK(J)
      J = J - 1
      K = K + 2
      GO TO 20
   25 JJ = IWK(J) + JJ
      J = 4
   30 ISP = IWK(K)
      IF (JJ .EQ. 0) GO TO 40
C                                   RESET TRIGONOMETRIC PARAMETERS
      C2 = JJ * ISP * RAD
      C1 = DCOS(C2)
      S1 = DSIN(C2)
   35 C2 = C1 * C1 - S1 * S1
      S2 = C1 * (S1 + S1)
      C3 = C2 * C1 - S2 * S1
      S3 = C2 * S1 + S2 * C1
```

```
   40 JSP = ISP + KB
C                                     DETERMINE FOURIER COEFFICIENTS
C                                       IN GROUPS OF 4
      DO 50 I=1,ISP
         K0 = JSP - I
         K1 = K0 + ISP
         K2 = K1 + ISP
         K3 = K2 + ISP
         ZA0 = A(K0)
         ZA1 = A(K1)
         ZA2 = A(K2)
         ZA3 = A(K3)
         IF (S1 .EQ. ZERO) GO TO 45
         TEMP = A1
         A1 = A1 * C1 - B1 * S1
         B1 = TEMP * S1 + B1 * C1
         TEMP = A2
         A2 = A2 * C2 - B2 * S2
         B2 = TEMP * S2 + B2 * C2
         TEMP = A3
         A3 = A3 * C3 - B3 * S3
         B3 = TEMP * S3 + B3 * C3
   45    TEMP = A0 + A2
         A2 = A0 - A2
         A0 = TEMP
         TEMP = A1 + A3
         A3 = A1 - A3
         A1 = TEMP
         TEMP = B0 + B2
         B2 = B0 - B2
         B0 = TEMP
         TEMP = B1 + B3
         B3 = B1 - B3
         B1 = TEMP
         A(K0) = DCMPLX(A0+A1,B0+B1)
         A(K1) = DCMPLX(A0-A1,B0-B1)
         A(K2) = DCMPLX(A2-B3,B2+A3)
         A(K3) = DCMPLX(A2+B3,B2-A3)
   50 CONTINUE
      IF (K .LE. 1) GO TO 55
      K = K - 2
      GO TO 30
   55 KB = K3 + ISP
C                                     CHECK FOR COMPLETION OF FINAL
C                                       ITERATION
      IF (KN .LE. KB) GO TO 70
      IF (J .NE. 1) GO TO 60
      K = 3
      J = MK
      GO TO 20
   60 J = J - 1
      C2 = C1
      IF (J .NE. 2) GO TO 65
      C1 = C1 * CK + S1 * SK
      S1 = S1 * CK - C2 * SK
      GO TO 35
   65 C1 = (C1 - S1) * SQ
      S1 = (C2 + S1) * SQ
      GO TO 35
   70 CONTINUE
```

```
C                                          PERMUTE THE COMPLEX VECTOR IN
C                                            REVERSE BINARY ORDER TO NORMAL
C                                            ORDER
      IF(M .LE. 1) GO TO 9005
      MP = M+1
      JJ = 1
C                                          INITIALIZE WORK VECTOR
      IWK(1) = 1
      DO 75  I = 2,MP
         IWK(I) = IWK(I-1) * 2
   75 CONTINUE
      N4 = IWK(MP-2)
      IF (M .GT. 2) N8 = IWK(MP-3)
      N2 = IWK(MP-1)
      LM = N2
      NN = IWK(MP)+1
      MP = MP-4
C                                          DETERMINE INDICES AND SWITCH A
      J = 2
   80 JK = JJ + N2
      AK2 = A(J)
      A(J) = A(JK)
      A(JK) = AK2
      J = J+1
      IF (JJ .GT. N4) GO TO 85
      JJ = JJ + N4
      GO TO 105
   85 JJ = JJ - N4
      IF (JJ .GT. N8) GO TO 90
      JJ = JJ + N8
      GO TO 105
   90 JJ = JJ - N8
      K = MP
   95 IF (IWK(K) .GE. JJ) GO TO 100
      JJ = JJ - IWK(K)
      K = K - 1
      GO TO 95
  100 JJ = IWK(K) + JJ
  105 IF (JJ .LE. J) GO TO 110
      K = NN - J
      JK = NN - JJ
      AK2 = A(J)
      A(J) = A(JJ)
      A(JJ) = AK2
      AK2 = A(K)
      A(K) = A(JK)
      A(JK) = AK2
  110 J = J + 1
C                                          CYCLE REPEATED UNTIL LIMITING NUMBER
C                                            OF CHANGES IS ACHIEVED
      IF (J .LE. LM) GO TO 80
C
 9005 CONTINUE

c
c  re-order the spectrum so that it runs from most-negative
c  to most-positive, with dc in the middle, and positive
c  defined as increasing phase with time.
c
      iflip = 0
```

```
      ipa = N/2
      ipb = 1
      do 10001 ip = ipa,ipb,-1
      iflip = iflip + 1
      sort(iflip) = A(ip)
10001 continue

      iflip = N/2
      ipa = N
      ipb = N/2+1
      do 10002 ip = ipa,ipb,-1
      iflip = iflip+1
      sort(iflip) = A(ip)
10002 continue

      ipb = N
      do 10003 ip = 1,ipb
      A(ip) = sort(ip)
10003 continue
      RETURN
      END
```

```
c
c
     Subroutine Header
c
c  This subroutine reads "Header.dat", an ascii file
c  containing the radar parameters.  The time of the
c  sounding is obtained from the name of the data file
c  "datafile".  Since time is the only parameter that
c  changes from sounding to sounding, this eliminates
c  handling a separate header file for each data file.
c
c  March 10, 1991
c
$Include:'Header.inc'
     character*1 thing1,thing2
c    write (*,*) datafile
     iunit = 88
20001 open (iunit,file='Header.dat',iostat=iocheck,status='old')
     if (iocheck .gt. 0) then
     iunit = iunit+1
     go to 20001
     endif

     read (iunit,*) SoundingNumber
     read (iunit,*) Year
     read (iunit,*) Month
     read (iunit,*) Day
     read (iunit,*) Hour
     read (iunit,*) Minute
     read (iunit,*) Second
     read (iunit,'(A)') SiteName
     read (iunit,'(A)') DataType
     read (iunit,'(A)') DataMode
     read (iunit,*) FFTpts
     read (iunit,*) Freq1
     read (iunit,*) Freq2
     read (iunit,*) PulseDuration
     read (iunit,*) FFTPeriod
     read (iunit,*) NumCohAve
     read (iunit,*) NumRangeGates
     read (iunit,*) NumRx
     read (iunit,*) PRP
     read (iunit,*) AltMin
     read (iunit,*) AltStep
     read (iunit,*) RxAttn
     read (iunit,'(A)') RxPolarization
     read (iunit,*) TxPower
     read (iunit,*) (RxMask(i),i=1,NumRx)
     read (iunit,'(A)') TapeLabel
     close (iunit)
c
c  Get the real time from the name of the data file.
c
     do 10001 iplace = 1,24
     if (datafile(iplace:iplace) .eq. '.') then
     imark = iplace
     go to 20002
10001 endif
     write (*,*) 'No . found in datafile name.'
     return
```

```
20002 do 10002 iplace = imark-8,imark-2,2
      thing1 = datafile(iplace:iplace)
      thing2 = datafile(iplace+1:iplace+1)
      if (thing1 .eq. '0') then
      i1 = 0
      elseif (thing1 .eq. '1') then
      i1 = 1
      elseif (thing1 .eq. '2') then
      i1 = 2
      elseif (thing1 .eq. '3') then
      i1 = 3
      elseif (thing1 .eq. '4') then
      i1 = 4
      elseif (thing1 .eq. '5') then
      i1 = 5
      elseif (thing1 .eq. '6') then
      i1 = 6
      elseif (thing1 .eq. '7') then
      i1 = 7
      elseif (thing1 .eq. '8') then
      i1 = 8
      elseif (thing1 .eq. '9') then
      i1 = 9
      else
      i1 = 0
      endif
      if (thing2 .eq. '0') then
      i2 = 0
      elseif (thing2 .eq. '1') then
      i2 = 1
      elseif (thing2 .eq. '2') then
      i2 = 2
      elseif (thing2 .eq. '3') then
      i2 = 3
      elseif (thing2 .eq. '4') then
      i2 = 4
      elseif (thing2 .eq. '5') then
      i2 = 5
      elseif (thing2 .eq. '6') then
      i2 = 6
      elseif (thing2 .eq. '7') then
      i2 = 7
      elseif (thing2 .eq. '8') then
      i2 = 8
      elseif (thing2 .eq. '9') then
      i2 = 9
      else
      i2 = 0
      endif

      if (iplace .eq. imark-8) Month = 10*i1 + i2
      if (iplace .eq. imark-6) Day = 10*i1 + i2
      if (iplace .eq. imark-4) Hour = 10*i1 + i2
      if (iplace .eq. imark-2) Minute = 10*i1 + i2
10002 continue

      thing1 = datafile(imark+1:imark+1)
      thing2 = datafile(imark+2:imark+2)
      if (thing1 .eq. '0') then
```

```
il = 0
elseif (thing1 .eq. '1') then
il = 1
elseif (thing1 .eq. '2') then
il = 2
elseif (thing1 .eq. '3') then
il = 3
elseif (thing1 .eq. '4') then
il = 4
elseif (thing1 .eq. '5') then
il = 5
elseif (thing1 .eq. '6') then
il = 6
elseif (thing1 .eq. '7') then
il = 7
elseif (thing1 .eq. '8') then
il = 8
elseif (thing1 .eq. '9') then
il = 9
else
il = 0
endif
if (thing2 .eq. '0') then
i2 = 0
elseif (thing2 .eq. '1') then
i2 = 1
elseif (thing2 .eq. '2') then
i2 = 2
elseif (thing2 .eq. '3') then
i2 = 3
elseif (thing2 .eq. '4') then
i2 = 4
elseif (thing2 .eq. '5') then
i2 = 5
elseif (thing2 .eq. '6') then
i2 = 6
elseif (thing2 .eq. '7') then
i2 = 7
elseif (thing2 .eq. '8') then
i2 = 8
elseif (thing2 .eq. '9') then
i2 = 9
else
i2 = 0
endif
Second = 10*il + i2

return
end
```

```
c
c
      Subroutine BTESTM
c
c  This subroutine applies the interferometry algorithms.
c
c  May 7, 1990
c
$include:'BsppM.inc'
$include:'Header.inc'

      complex xx,SumVolt,VA,VB,ordinary

      do 10099 dir = 1,2
      ZASpread(dir) = 0
      do 10098 dipole = 1,2
      ZAW = 3*ZAWindow - ZASpread(dir)
      do 10004 ant = 1,3
c
c  Test #1: Reject this Doppler frequency if both quadrature
c  components are too small on any antenna for an accurate calculation
c  of the phase.  This happens not too often.
c
      if (abs(xData(dir,ant,dipole,dopp,1)) .le. 10 .and.
     1    abs(xData(dir,ant,dipole,dopp,2)) .le. 10) then
      reject(1) = reject(1) + 1
      failflag = 1
      go to 90909
      endif
c
c  calculate the phase.
c
      Phase(dir,ant,dipole) =
     1          ATan2(xData(dir,ant,dipole,dopp,2),
     2                xData(dir,ant,dipole,dopp,1))
10004 continue

c
c  Calculate the antenna-to-antenna phase differences.
c
      pd12(dir,dipole) = Phase(dir,2,dipole)-Phase(dir,1,dipole)

      If (pd12(dir,dipole) .gt. pi)
     1    pd12(dir,dipole) = pd12(dir,dipole) - 2*pi
      If (pd12(dir,dipole) .lt. -pi)
     1    pd12(dir,dipole) = pd12(dir,dipole) + 2*pi

      pd23(dir,dipole) = Phase(dir,3,dipole)-Phase(dir,2,dipole)
      If (pd23(dir,dipole) .gt. pi)
     1    pd23(dir,dipole) = pd23(dir,dipole) - 2*pi
      If (pd23(dir,dipole) .lt. -pi)
     1    pd23(dir,dipole) = pd23(dir,dipole) + 2*pi

c
c  Tests #2,3,6,&7: The two zenith angles derived from the two phase
c  differences for each dipole, each direction, must agree.
c
c  Each time through, the maximum allowed
c  zenith angle spread, ZAW, is 3*ZAWindow - ZASpread(dir).
c  The actual maxima are accumulated in ZASpread(dir).
```

```
c  Each time through, the actual spread is added to ZASpread(dir),
c  which is the only measure of the zenith-angle spread that is
c  saved as part of the scattering-point parameters.
c
c   Antenna Pair 1-2:
c
      theta1 = asin(-pd12(dir,dipole)/pi)
c
c  Antenna Pair 2-3:
c
      theta2 = asin(-pd23(dir,dipole)/pi)
c
c  Are the two zenith angles close enough together to
c  qualify as a scattering point?
c  If thSpread is greater than the maximum spread allowed (ZAW),
c  set failflag=1 and get out.  Otherwise, possible scattering point.
c  This gives reject(2,3,6,&7).
c
      thSpread = abs(theta1-theta2)
      if (thSpread .gt. ZAW) then
      index = (dir-1)*4 + dipole + 1
      reject(index) = reject(index) + 1
      failflag = 1
      go to 90909
      endif

c
c  Possible scattering point; we've found an acceptably small
c  disagreement (thSpread) between the two theta values.
c  Accumulate thSpread in ZASpread(dir) (which will become two of the
c  scattering-point parameters), and locate the average zenith angle
c  for this dir and dipole at the middle of the window.
c
      ZASpread(dir) = ZASpread(dir) + thSpread
      th1Dipole(dipole) = (theta1+theta2)/2
10098 continue
c
c  Tests #4 and #8: Both dipoles have separately determined zenith
c  angles for one direction.  Do these two values agree?
c
      ZAW = 3*ZAWindow - ZASpread(dir)
      if (abs(th1Dipole(1)-th1Dipole(2)) .gt. ZAW) then
      index = 4*dir
      reject(index) = reject(index) + 1
      failflag = 1
      go to 90909
      endif

      thConsensus = (th1Dipole(1)+th1Dipole(2))/2
      thSpread = abs(th1Dipole(1)-th1Dipole(2))
      ZASpread(dir) = ZASpread(dir) + thSpread
c
c  Now we are convinced that the two dipoles together
c  indicate that a real scattering point exist.  Before we
c  can test the two directions together, we need the best
c  possible estimate of the zenith angles.
c  We use the 1-3 antenna pair with the dipoles combined to match
c  the transmit polarization.
c
      if (TxPolarization .eq. 'O') then
```

```
      VA = cmplx(xdata(dir,1,1,dopp,1),xdata(dir,1,1,dopp,2))
     1   + srmo*cmplx(xdata(dir,1,2,dopp,1),xdata(dir,1,2,dopp,2))

      VB = cmplx(xdata(dir,3,1,dopp,1),xdata(dir,3,1,dopp,2))
     1   + srmo*cmplx(xdata(dir,3,2,dopp,1),xdata(dir,3,2,dopp,2))

      elseif (TxPolarization .eq. 'X') then
      VA = cmplx(xdata(dir,1,1,dopp,1),xdata(dir,1,1,dopp,2))
     1   - srmo*cmplx(xdata(dir,1,2,dopp,1),xdata(dir,1,2,dopp,2))

      VB = cmplx(xdata(dir,3,1,dopp,1),xdata(dir,3,1,dopp,2))
     1   - srmo*cmplx(xdata(dir,3,2,dopp,1),xdata(dir,3,2,dopp,2))

      elseif (TxPolarization .eq. 'L') then
      VA = cmplx(xdata(dir,1,1,dopp,1),xdata(dir,1,1,dopp,2))
     1   + cmplx(xdata(dir,1,2,dopp,1),xdata(dir,1,2,dopp,2))

      VB = cmplx(xdata(dir,3,1,dopp,1),xdata(dir,3,1,dopp,2))
     1   + cmplx(xdata(dir,3,2,dopp,1),xdata(dir,3,2,dopp,2))

      else
      write (*,*) 'TxPolarization = ',TxPolarization
      write (*,*) 'is NOT SUPPORTED by this version of SPPM'
      go to 90909
      endif
c
c  Antenna Pair A-B (= 1-3 with polarization):
c
      VAr = real(VA)
      VAi = aimag(VA)
      VBr = real(VB)
      VBi = aimag(VB)

      if ((abs(VAr) .gt. 1e-5) .or. (abs(VAi) .gt. 1e-5)) then
      PhaseA = Atan2(VAi,VAr)
      else
      PhaseA = 0
      endif

      if ((abs(VBr) .gt. 1e-5) .or. (abs(VBi) .gt. 1e-5)) then
      PhaseB = Atan2(VBi,VBr)
      else
      PhaseB = 0
      endif

      pdAB = PhaseB-PhaseA
      If (pdAB .gt. +pi) pdAB = pdAB - 2*pi
      If (pdAB .lt. -pi) pdAB = pdAB + 2*pi

      do 10007 ithree = -1,1
      sinTheta = -(pdAB+2*pi*ithree)/(2*pi)

      if (abs(sinTheta) .lt. 1.0) then
      thAB = asin(sinTheta)

      if (abs(thAB-thConsensus) .lt. ZAWindow) then
      thetafinal(dir) = thAB
      go to 10099

      endif
```

```
      endif

10007 continue

c
c  Test #5 and #9: If there is no final zenith angle in adequate
c  agreement with the Consensus zenith angle, then you fall through
c  to here and leave a failure.
c
      index = 5 + 4*(dir-1)
      reject(index) = reject(index) + 1
      failflag = 1
      go to 90909
c
10099 continue
c
c  Test #10: Reject if no real altitude is possible.
c
      arg =  sin(thetafinal(1))**2+sin(thetafinal(2))**2
      if (arg .ge. 1) then
      reject(10) = reject(10) + 1
      failflag = 1
      go to 90909
      endif
c
c  If you got to here, it's a real scattering point, and the
c  cardinal zenith angles are:
c
      ThetaEW = thetafinal(1)
      ThetaNS = thetafinal(2)
c
c  these will get rotated by 45 degrees in the main program if
c  DataMode = 3.
c
90909 return
      end
```

```
c
c
      Subroutine BSTEERM
c
c  Now that you've found the point, "steer" the full array toward the
c  point and determine its power and phase.
c  May 7, 1990.
c
$include:'BsppM.inc'
$include:'Header.inc'

      complex Vsteer(2)
c
c  Each direction and dipole is steered separately, and the voltage
c  amplitudes and phases determined.
c

      do 10002 dipole = 1,2
      Vsteer(dipole) = 0
      do 10001 dir = 1,2
      delphi(dir) = 2*pi*sin(thetafinal(dir))
      do 10001 ant = 1,3
      if (ant .eq. 3 .and. dir .eq. 2) go to 10001
      Vsteer(dipole) = Vsteer(dipole)
     1 + cmplx(1.0*xdata(dir,ant,dipole,dopp,1),
     2           xdata(dir,ant,dipole,dopp,2))
     3 * cexp(+srmo*AntLocation(ant)*delphi(dir))
10001 continue

      Vx = Real(Vsteer(dipole))
      Vy = Aimag(Vsteer(dipole))

      VAmplitude(dipole) = cabs(Vsteer(dipole))

      if (VAmplitude(dipole) .ge. 1) then
      Faze(dipole) = ATan2(Vy,Vx)
      else
      Faze(dipole) = 0
      endif

10002 continue

90909 return
      end
```

```
c
$Debug
c
      Subroutine BSORTM
c
c  The scattering-point parameters are in spptemp
c  in order by range, but only roughly by altitude.
c  This subroutine orders them by altitude and writes
c  them into the output file, SPPList.
c
$include:'Bsppm.inc'
$include:'Header.inc'

      real*4 zlast
      integer*4 itop,ipoint,ipmin
      isort = 1
      itop = sppnumber
20001 zmin = 999
      zlast = 0
c
c  Look through all the points to find the smallest altitude.
c
      do 10001 ipoint = 1,itop
      if (spptemp(ipoint,1) .gt. -990 .and.
     1    spptemp(ipoint,1) .lt. zmin) then
      zmin = spptemp(ipoint,1)
      ipmin = ipoint
      endif
10001 continue

      if (zmin .lt. zlast) WRITE (*,*) 'PROBLEM! ',zmin,ipmin,zlast
      zlast = zmin

      write (3) (spptemp(ipmin,parameter),parameter=1,10)
      if (itop .gt. 1) then
c
c  Replace the last scattering point by the one at the top.
c
      do 10004 parameter=1,10
      spptemp(ipmin,parameter) = spptemp(itop,parameter)
10004 continue
c
c  Repeat until they're all gone.  I didn't say it was efficient.
c
      itop = itop-1
      go to 20001
      endif
30001 return
      end
```

```
c
c  BsppM.inc
c
      common /sppA/ spptemp(2500,10)
      Common /sppB/ Data(2,3,2,256,2),Phase(2,3,2),theta(3,3),
     1              delphi(2),thetafinal(2),ZASpread(2),
     2              Voltage(2,2),VAmplitude(2),Power(2,2),Faze(2),
     3              pd12(2,2),pd23(2,2),reject(12),thlDipole(2),
     4              AntLocation(5),S(10),xdata(2,3,2,256,2)
      common /sppC/ dopp,failflag,jRange,rgl,rg2,TxPolarization,
     1              SumPower,SPPNumber,ThetaEW,ThetaNS,ZAWindow,
     2              pi,contod,contor,srmo,clight,Threshold,LPswitch
      Complex*16 Voltage,srmo
      Real*8 pi,xorig,yorig,xyangle,rotate,ThetaEW,ThetaNS
      Real*4 zmin,spptemp,Threshold,plimit,S,xdata
      Integer*4 Data,reject
      Integer*4 ant,count,dipole,dir,dopp,failflag,parameter,pmin,
     1         point,pulse,quad,rgl,rg2,spacing,SPPNumber
      Character*2 TxPolarization
```

## Header.dat

```
 938                     SoundingNumber
89                       Year
 5                       Month
 3                       Day
17                       Hour
31                       Minute
16                       Second
Islote, P.R.             SiteName
0001                     DataType
0003                     DataMode
 256                     FFTPts
 3.17500E+06             Freq1 (Hz)
 3.17500E+06             Freq2 (Hz)
 3.00000E-05             PulseDuration (seconds)
 1.02400E+02             FFTPeriod (seconds)
 4.00000E+01             NumCohAve
  40                     NumRangeGates
  10                     NumRx
 1.00000E-02             PRP (seconds)
+3.10000E+03             AltMin (meters)
 3.00000E+03             AltStep (meters)
   0                     RxAttn
L1                       RxPolarization
 1.00000E+05             TxPower (Watts)
1 1 1 1 1 1 1 1 1 1      RxMask
GRxxx                    TapeLabel
```

```
c
$Debug
      program WindErr
c
*******+************************************************************
*                                                                  *
*      IDI Wind-Calculation Program; MAPSTAR Radar.                *
*      Copyright 1990, Holodyne Limited 1986.                      *
*      All Rights Reserved.                                        *
*                                                                  *
********************************************************************
c   April 8, 1991
c
c   This program will calculate 129 wind profiles
c   for a single scattering-point parameter file.  The spread due to
c   errors in the calculation are determined by varying the range,
c   the Doppler velocity, and the E-W and N-S zenith angles.  The
c   129 profiles so generated are examined to get max and min at
c   each altitude.  Files generated are u.dat, ubar.dat, v.dat,
c   vbar.dat, w.dat, and wbar.dat, which are for plotting the
c   three components and their error bars, and winderr.dat, which
c   contains all the information in a single file.
c
c   The scattering-point parameters
c   are :
c   1.  Altitude (km).
c   2.  Radial velocity (m/sec).
c   3.  Zenith angle in East-West meridian (degrees).
c   4.  Zenith angle in North-South meridian (degrees).
c   5.  Voltage amplitude on #1 Dipoles.
c   6.  Phase of #1 Dipoles (degrees).
c   7.  Voltage amplitude on #2 Dipoles.
c   8.  Phase of #2 Dipoles (degrees).
c   9.  E-W zenith-angle spread.
c   10. N-S zenith-angle spread.
c
c   Explanation of easily-reprogrammed parameters (just change the source-
c      code value given below:
c   vHmax is the largest allowed horizontal velocity. We test each point
c     against Vmax by projecting its radial velocity into the horizontal
c     plane, and reject it if it's bigger than vHmax.
c   ThMaxV is the largest acceptable radial zenith angle for w.
c   ThMinV is the smallest acceptable radial zenith angle for w.
c   ThMaxH is the largest acceptable radial zenith angle for u and v.
c   ThMinH is the smallest acceptable radial zenith angle for u and v.
c   MinNumPts is the minimum number of points.  If there are not sufficient
c      points, that altitude is skipped.
c   NSigma is the maximum number of standard deviations from the fit any
c     individual point can lie without being rejected from the velocity
c     calculation.
c   Zmin is the bottom altitude for which winds are to be calculated.
c
c  WindErr calls SppFltr, Header, WFV, and WFH.
c
$Include:'Wind.Inc'
$Include:'Header.inc'
      dimension umax(100),u0(100),umin(100),
     1          vmax(100),v0(100),vmin(100),
     2          wmax(100),w0(100),wmin(100),scale(100),
     3          Number(100),Uvar(100),Vvar(100),Wvar(100)
```

```
      real*4 NumSndgs,scale
      integer*4 jzMax
      pi = 3.14159265

      vHmax = 300
      ThMinV = 0
      ThMaxV = 10
      ThMinH = 3
      ThMaxH = 16
      MinH = 5
      MinV = 5
      Nsigma = 3.0
      Zmin = 60
      Zmax = 120
      polarization = 'o'
************************************************************************
*
      Open (1,err=90909,file=' ',status='old',form='binary')
      Open (2,err=90909,file='SppFltr.mbs',form='binary')
      Call SppFltr
      close (1)
      rewind (2)
      CALL Head

      open (1,file='WE2.dat')

      do 10001 jz = 1,100
      Z(jz) =  float(jz-1) + Zmin
      Umax(jz) = -100
      Umin(jz) = +100
      u0(jz) = 0
      Uvar(jz) = 0
      Vmax(jz) = -100
      Vmin(jz) = +100
      v0(jz) = 0
      Vvar(jz) = 0
      Wmax(jz) = -10
      Wmin(jz) = +10
      w0(jz) = 0
      Wvar(jz) = 0
      Number(jz) = 0
      scale(jz) = 1
      if (Z(jz) .ge. Zmax) then
      jZmax = jz
      go to 20301
      endif
10001 continue


20301 do 10501 jZinc = 1,3
      if (jZinc .eq. 1) then
      write (*,*) '.'
      elseif (jZinc .eq. 2) then
      write (*,*) '..'
      else
      write (*,*) '...'
      endif
      write (*,90008)
90008 format (1x,\)
      do 10301 idR = 0,2
```

```
      dR = idR*2.5
      if (idR .eq. 2) dR = -2.5
      do 10301 idVr = 0,4
      if (idVr .eq. 0) dVrad =  0
      if (idVr .eq. 1) dVrad = +0.23
      if (idVr .eq. 2) dVrad = -0.23
      if (idVr  eq. 3) dVrad = +0.23
      if (idVr .eq. 4) dVrad = -0.23
      do 10301 idThEW = 0,4
      do 10301 idThNS = 0,4
      rewind (2)
      iSum = idR + idVr + idThEW + idThNS
      if (((idR .eq. 0) .or. (idVr .eq. 0) .or.
     1     (idThEW .eq. 0) .or. (idThNS .eq. 0)) .and.
     2      iSum .gt. 0) go to 10301

      index = (idVr-1)*16 + (idThEW-1)*4 + idThNS
      if (index/64 .lt. 1) then
      write (*,90005)
90005 format ('.',\)
      else
      write (*,90006)
90006 format ('.')
      write (*,90007)
90007 format (1x,\)
      endif

      QuitFlag = 0
      jzW = jZinc
      do 10101 jz = 1,jZmax
      u(jz) = 0
      v(jz) = 0
      w(jz) = 0
10101 continue
      read (2,err=90909,end=20203) (line(parameter),parameter=1,10)

*************************************************************************
* Return to here for new altitude.
*************************************************************************
c
20201 NumPts = 0
      read (2,err=90909,end=20203) (line(parameter),parameter=1,10)
20202 if (line(1)+dR*scale(jzW) .le. Z(jzW)-1.5) then
      read (2,err=90909,end=20203) (line(parameter),parameter=1,10)
      go to 20202
      endif
      if (line(1)+dR*scale(jzW) .gt. Z(jzW)+1.5) then
      if ((NumPts .lt. MinV) .or. (NumPts .lt. MinH)) go to 20206
      go to 20204
      endif

      TestFlag = 1
      if (NumPts .eq. 5000) then
      write (*,*) 'Thanks anyhow, but Ive already got 5000 points.'
      TestFlag = 0
      endif

      if (TestFlag .eq. 1) then
      NumPts = NumPts + 1
```

```
      do 10201 parameter = 1,10
      if (parameter .eq. 2) then
      if (idVr .le  2) then
      if (line(2) .lt. 0) spp(NumPts,2) = line(2) - dVrad*scale(jzW)
      if (line(2) .gt. 0) spp(NumPts,2) = line(2) + dVrad*scale(jzW)
      else
      spp(NumPts,2) = line(2) + dVrad*scale(jzW)
      endif

      elseif (parameter .eq. 3) then
      if (idThEW .eq. 0) then
      spp(NumPts,3) = line(3)
      elseif (idThEW .eq. 1) then
      if (line(3) .lt. 0) spp(NumPts,3) = line(3) - line(9)*scale(jzW)/2
      if (line(3) .ge. 0) spp(NumPts,3) = line(3) + line(9)*scale(jzW)/2
      elseif (idThEW .eq. 2) then
      if (line(3) .lt. 0) spp(NumPts,3) = line(3) + line(9)*scale(jzW)/2
      if (line(3) .ge. 0) spp(NumPts,3) = line(3) - line(9)*scale(jzW)/2
      elseif (idThEW .eq. 3) then
      spp(Numpts,3) = line(3) + line(9)*scale(jzW)/2
      elseif (idThEW .eq. 4) then
      spp(NumPts,3) = line(3) - line(9)*scale(jzW)/2
      endif

      elseif (parameter .eq. 4) then
      if (idThNS .eq. 0) then
      spp(NumPts,4) = line(4)
      elseif (idThNS .eq. 1) then
      if (line(4) .lt. 0) spp(NumPts,4) = line(4)-line(10)*scale(jzW)/2
      if (line(4) .ge. 0) spp(NumPts,4) = line(4)+line(10)*scale(jzW)/2
      elseif (idThNS .eq. 2) then
      if (line(4) .lt. 0) spp(NumPts,4) = line(4)+line(10)*scale(jzW)/2
      if (line(4) .ge. 0) spp(NumPts,4) = line(4)-line(10)*scale(jzW)/2
      elseif (idThNS .eq. 3) then
      spp(Numpts,4) = line(4) + line(10)*scale(jzW)/2
      elseif (idThNS .eq. 4) then
      spp(NumPts,4) = line(4) - line(10)*scale(jzW)/2
      endif

      else
      spp(NumPts,parameter) = line(parameter)
      endif
10201 continue

      endif

      read (2,err=90909,end=20203) (line(parameter),parameter=1,10)
      go to 20202

20203 quitflag = 1

20204 Fitflag = 1
c
c  Fit the scattering points in this window with a 3-vector.
c
20205 CALL WFV
      if (Fitflag .eq. 0) then
c     write (*,*) 'Vertical Failure at ',jzW,Z(jzW)
      go to 20206
      endif
```

```
      Call WFH

      if (Fitflag .eq. 0) then
c     write (*,*) 'Horizontal Failure at ',jzW,Z(jzW)
      else
      if ((idR .eq. 0) .and. (idVr .eq. 0) .and.
     1    (idThEW .eq. 0) .and. (idThNS .eq. 0)) then
      if (NumPts .eq. 0) then
      scale(jzW) = 1
      else
      scale(jzW) = 1/sqrt( float(NumPts)/3.0)
      endif
      endif

c     write (*,90001) idR,idVr,idThEW,idThNS,Z(jzW),u(jzW),v(jzW),w(jzW)
      write (1,90004) Z(jzW),u(jzW),v(jzW),w(jzW)
      endif
90001 format (1x,4(i2,1x),f4.0,2(1x,f6.1),1x,f5.1)
90004 format (1x,f4.0,2(1x,f6.1),1x,f5.1)
90002 format (1x,4(e12.4,1x))
c
c  If it's not time to quit, increment jzW and go read the next points.
c
20206 if (QuitFlag .eq. 0) then
      jzW = jzW + 3
      if (jzW .gt. jZmax) go to 10301
      read (2,err=90909,end=20203) (line(parameter),parameter=1,10)
      go to 20201
      endif

10301 continue
10501 continue
90909 close (2)

      rewind (1)

20401 read (1,90004,end=20402) Zx,ux,vx,wx
      ijz = Zx-Zmin+1
      U0(ijz) = U0(ijz) + ux
      V0(ijz) = V0(ijz) + vx
      W0(ijz) = W0(ijz) + wx
      Number(ijz) = Number(ijz) + 1
      go to 20401

20402 do 10601 jz = 1,jzMax
      if (Number(jz) .gt. 0) then
      U0(jz) = U0(jz)/Number(jz)
      V0(jz) = V0(jz)/Number(jz)
      W0(jz) = W0(jz)/Number(jz)
      endif
10601 continue

      rewind (1)

20403 read (1,90004,end=20404) Zx,ux,vx,wx
      ijz = Zx-Zmin+1
      Uvar(ijz) = Uvar(ijz) + abs(u0(ijz)-ux)
      Vvar(ijz) = Vvar(ijz) + abs(v0(ijz)-vx)
      Wvar(ijz) = Wvar(ijz) + abs(w0(ijz)-wx)
```

```
      go to 20403

20404 do 10602 jz = 1,jzMax
      if (Number(jz) .gt. 0) then
      Uvar(jz) = Uvar(jz)/Number(jz)
      Vvar(jz) = Vvar(jz)/Number(jz)
      Wvar(jz) = Wvar(jz)/Number(jz)
      endif
10602 continue

20405 CLOSE (1)

      open (1,file='u.dat')
      open (2,file='ubar.dat')
      do 10401 jz = 1,jzMax
      if (Number(jz) .gt. 0) then
      umin(jz) = u0(jz) - 2*uvar(jz)
      umax(jz) = u0(jz) + 2*uvar(jz)
      if (umin(jz) .lt. -100) umin(jz) = -100
      if (umax(jz) .gt. +100) umax(jz) = +100
      write (1,*) z(jz),u0(jz)
      write (2,*) z(jz),umin(jz)
      write (2,*) z(jz),umax(jz)
      write (2,*) -10,0
      else
      write (1,*) -10,0
      endif
10401 continue
      close (1)
      close (2)

      open (1,file='v.dat')
      open (2,file='vbar.dat')
      do 10402 jz=1,jzMax
      if (Number(jz) .gt. 0) then
      vmin(jz) = v0(jz) - 2[illegible]ar(jz)
      vmax(jz) = v0(jz) + 2*v[illegible](jz)
      if (vmin(jz) .lt. -100) vm[illegible] = -100
      if (vmax(jz) .gt. +100) vmax[illegible] +100
      write (1,*) z(jz),v0(jz)
      write (2,*) z(jz),vmin(jz)
      write (2,*) z(jz),vmax(jz)
      write (2,*) -10,0
      else
      write (1,*) -10,0
      endif
10402 continue
      close (1)
      close (2)

      open (1,file='w.dat')
      open (2,file='wbar.dat')
      open (11,file='winderr.dat')

      do 10403 jz=1,jzMax
      if (Number(jz) .gt. 0) then
      wmin(jz) = w0(jz) - 2*wvar(jz)
      wmax(jz) = w0(jz) + 2*wvar(jz)
      if (wmin(jz) .lt. -10) wmin(jz) = -10
      if (wmax(jz) .gt. +10) wmax(jz) = +10
```

```
      write (1,*) z(jz),w0(jz)*10
      write (2,*) z(jz),wmin(jz)*10
      write (2,*) z(jz),wmax(jz)*10
      write (2,*) 0,-10

      if (z(jz) .ge. Zmin) then
      write (11,90003) z(jz),
     1        umin(jz),u0(jz),umax(jz),
     2        vmin(jz),v0(jz),vmax(jz),
     3        wmin(jz),w0(jz),wmax(jz)
      endif
      else
      write (1,*) -10,0
      endif
10403 continue
      close (1)
      close (2)
      close (11)
90003 format (1x,10(f7.2))
c      call BellSub
91919 End
```

```
c
$Debug
c
      subroutine SppFltr
      dimension spp(10)
      real*4 Zmin,Zmax,ThMin,ThMax,ZA
      character*1 polarization
      integer*4 parameter,Nraw,Nfltr
      pi = 3.14159265
      write (*,*) 'SppFltr expects the input (.mbs) file to be on the co
     lmmand line.'
      write (*,*)
     1 'The filtered file will be written to SppFltr.mbs.'
      Zmin = 60
      Zmax = 120
      ThMin = 0
      ThMax = 16
      VRmin = -60
      VRmax = 60
      vHmax = 300
      Nraw = 0
      Nfltr = 0
      polarization = 'o'
20001 read (1,end=20002) (spp(parameter),parameter=1,10)
      Nraw = Nraw + 1
**********************************************************************
c               Filter Section
c
c  Filters go here.  If fail, go to 20001.
c
      if (spp(1) .lt. zmin .or. spp(1) .gt. zmax) go to 20001
      if (spp(2) .eq. 0) go to 20001
      if (spp(2) .lt. VRmin .or. spp(2) .gt. VRmax) go to 20001
      sinZA = sqrt(sin(spp(3)*pi/180)**2+sin(spp(4)*pi/180)**2)
      if (sinZA .ge. 1) go to 20001
      if (sinZA .gt. 1e-3) then
      if (abs(spp(2)/sinZA) .gt. vHmax) go to 20001
      endif
      ZA = (180/pi)*asin(sinZA)
      if ((ZA .lt. ThMin) .or. (ZA .gt. ThMax)) go to 20001
c
c  Linear polarization filter (removes linearly polarized points):
c
      if ((polarization .eq. 'o') .or. (polarization .eq. 'x') .or.
     1     (polarization .eq. 'c')) then
      if (abs(spp(6)-spp(8)) .lt. 45)  go to 20001
      if ( (abs(spp(6)-spp(8)) .gt. 135) .and.
     1       (abs(spp(6)-spp(8)) .lt. 225) ) go to 20001
      if ( (abs(spp(6)-spp(8)) .gt. 315) .and.
     1       (abs(spp(6)-spp(8)) .lt. 360) ) go to 20001
      endif
c
c
c  Ordinary polarization filter (removes ordinary points):
c
      if ((polarization .eq. 'l') .or. (polarization .eq. 'x')) then
      if (spp(6)-spp(8) .gt. 45  .and.
     1    spp(6)-spp(8) .lt. 135 )  go to 20001
      if (spp(6)-spp(8) .gt. -315  .and.
     1    spp(6)-spp(8) .lt. -225 )  go to 20001
```

```
      endif
c
c
c  Extraordinary polarization filter (removes extraordinary points):
c
      if ((polarization .eq. 'l') .or. (polarization .eq. 'o')) then
      if (spp(6)-spp(8) .gt. -135  .and.
     1    spp(6)-spp(8) .lt. -45 )  go to 20001
      if (spp(6)-spp(8) .gt. 225  .and.
     1    spp(6)-spp(8) .lt. 315 )  go to 20001
      endif
c
***********************************************************************

      Nfltr = Nfltr + 1
      write (2) (spp(parameter),parameter=1,10)
      go to 20001
20002 write (*,*) 'Points in, points out = ',Nraw,Nfltr
      return
      end
```

```
c
$Debug
c
**********************************************************************
      Subroutine WFV
**********************************************************************
C
C   THIS SUBROUTINE CALCULATES the vertical WINDS FROM MAPSTAR SPPs.
C   August 17, 1990
c
$Include:'Wind.Inc'
$Include:'Header.inc'

      Dimension a(3,3),WindV(3)
      Real*4 Sigma,SigmaLast
      integer*4 flag,iZA
      pi = 3.14159265
      Do 10101 Ia = 1,3
      WindV(ia) = 0
      Do 10101 ib = 1,3
      A(ia,ib) = 0
10101 Continue
      NPV = NumPts
      do 10201 point = 1,NumPts
      iwtv(point) = 1
      sinZA(point) = sqrt(sin(spp(point,3)*pi/180)**2
     1        +       sin(spp(point,4)*pi/180)**2)
      if ((sinZA(point) .lt. sin(ThMinV*pi/180)) .or.
     1    (sinZA(point) .gt. sin(ThMaxV*pi/180))) then
      iwtv(point) = 0
      NPV = NPV - 1
      if (NPV .lt. MinV) then
      FitFlag = 0
      go to 90909
      endif
      endif
      CosL(point) = Sin(spp(point,3)*pi/180)
      CosM(point) = Sin(spp(point,4)*pi/180)
      CosN(point) = sqrt(1 - CosL(point)**2 - CosM(point)**2)
10201 continue
      SigmaLast = 1e8
20001 flag = 0
      Do 10301 point = 1,NumPts
      if (iwtv(point) .eq. 0) go to 10301
      A(1,1) = A(1,1) + CosL(point)**2
      A(1,2) = A(1,2) + CosL(point)*CosM(point)
      A(1,3) = A(1,3) + CosL(point)*CosN(point)
      A(2,2) = A(2,2) + CosM(point)**2
      A(2,3) = A(2,3) + CosM(point)*CosN(point)
      A(3,3) = A(3,3) + CosN(point)**2
      WindV(1) = WindV(1) + SPP(point,2)*CosL(point)
      WindV(2) = WindV(2) + SPP(point,2)*CosM(point)
      WindV(3) = WindV(3) + SPP(point,2)*CosN(point)
10301 Continue
      A(2,1) = A(1,2)
      A(3,1) = A(1,3)
      A(3,2) = A(2,3)
      det = a(1,1)*a(2,2)*a(3,3) + 2*a(1,2)*a(1,3)*a(2,3) -
     1      a(1,1)*a(2,3)**2 - a(2,2)*a(1,3)**2 - a(3,3)*a(1,2)**2
```

```
      If (abs(det) .lt. 1.0e-7) then
      write (*,*) 'WFV: no solution'
      Fitflag = 0
      go to 90909
      endif

      u(jzW) = (WindV(1)*(a(2,2)*a(3,3)-  a(2,3)**2) +
     1     WindV(2)*(a(2,3)*a(1,3) - a(1,2)*a(3,3)) +
     2     WindV(3)*(a(1,2)*a(2,3) - a(1,3)*a(2,2)))/det
      v(jzW) = (WindV(1)*(a(2,3)*a(1,3) - a(1,2)*a(3,3)) +
     1      WindV(2)*(a(1,1)*a(3,3) - a(1,3)**2) +
     2      WindV(3)*(a(1,3)*a(1,2) - a(1,1)*a(2,3)))/det
      w(jzW) = (WindV(1)*(a(1,2)*a(2,3) - a(1,3)*a(2,2)) +
     1      WindV(2)*(a(1,2)*a(1,3) - a(1,1)*a(2,3)) +
     2      WindV(3)*(a(1,1)*a(2,2) - a(1,2)**2))/det
c
c  Calculate the Standard Deviation (Sigma)
c
      ErrorSum = 0
      Do 10401 point = 1,NumPts
      if (iwtv(point) .eq. 0) go to 10401
      dvr(point) = spp(point,2) - u(jzW)*CosL(point)
     1           - v(jzW)*CosM(point) - w(jzW)*CosN(point)
      ErrorSum = ErrorSum + dvr(point)**2
10401 Continue
      Sigma = sqrt(ErrorSum/NPV)

      Do 10501 point = 1,NumPts
      if (iwtv(point) .eq. 0) go to 10501
      if (abs(dvr(point)) .gt. NSigma*Sigma) then
      iwtv(point) = 0
      flag = 1
      NPV = NPV - 1
      if (NPV .lt. MinV) then
      FitFlag = 0
      go to 90909
      endif
      endif
10501 Continue

      if (flag .eq. 0) go to 20002
      if (flag .eq. 1) then
      if (Sigma .ge. 0.999*SigmaLast) go to 20002
      if (Sigma .le. 0.01) go to 20002
      SigmaLast = Sigma
      go to 20001
      endif
c
c  good velocity.
c
20002 if ( (abs(u(jzW)) .gt. vHmax)      .or.
     1     (abs(v(jzW)) .gt. vHmax)      .or.
     2     (abs(w(jzW)) .gt. vHmax/20) ) then
c     write (*,*) 'jzw,NumPts,u,v,w = '
c     write (*,*) jzw,NumPts,u(jzw),v(jzw),w(jzw)
      FitFlag = 0
      go to 90909
      endif

90909 Return
```

End

```
c
$Debug
c
*************************************************************************
      Subroutine WFH
*************************************************************************
C
C   THIS SUBROUTINE CALCULATES horizontal WINDS FROM MAPSTAR SPPs.
C   August 17, 1990
c
$Include:'Wind.Inc'
$Include:'Header.inc'

      Dimension H(3,3),Wind(3)
      Real*4 Sigma,SigmaLast
      integer*4 flag,iZA
      pi = 3.14159265
      Do 10101 Ia = 1,3
      Wind(ia) = 0
      Do 10101 ib = 1,3
      H(ia,ib) = 0
10101 Continue
      do 10102 iDir = 1,3
      PPCnt(iDir) = 0
      do 10102 ii = 1,17
      Numrad(ii,iDir) = 0
      vrad(ii,iDir) = 0
10102 Continue
      NPH = NumPts
      do 10201 point = 1,NumPts
      iwth(point) = 1
      if ((sinZA(point) .lt. sin(ThMinH*pi/180)) .or.
     1    (sinZA(point) .gt. sin(ThMaxH*pi/180))) then
      iwth(point) = 0
      NPH = NPH - 1
      if (NPH .lt. MinH) then
      FitFlag = 0
      go to 90909
      endif
      endif
      CosL(point) = Sin(spp(point,3)*pi/180)
      CosM(point) = Sin(spp(point,4)*pi/180)
      CosN(point) = sqrt(1 - CosL(point)**2 - CosM(point)**2)
10201 continue
      SigmaLast = 1e8
20001 flag = 0
      Do 10301 point = 1,NumPts
      if (iwth(point) .eq. 0) go to 10301
      H(1,1) = H(1,1) + CosL(point)**2
      H(1,2) = H(1,2) + CosL(point)*CosM(point)
      H(2,2) = H(2,2) + CosM(point)**2
      Wind(1) = Wind(1) + SPP(point,2)*CosL(point)
     1                  - CosL(point)*w(jzW)
      Wind(2) = Wind(2) + SPP(point,2)*CosM(point)
     1                  - CosM(point)*w(jzW)
10301 Continue
      H(2,1) = H(1,2)
      det = H(1,1)*H(2,2) - H(1,2)**2

      If (abs(det) .lt. 1.0e-7) then
```

```
      write (*,*) 'MVH: no solution'
      Fitflag = 0
      go to 90909
      endif

      u(jzW) = (wind(1)*H(2,2) - wind(2)*H(1,2))/det
      v(jzW) = (H(1,1)*wind(2) - H(1,2)*wind(1))/det
c
c  Calculate the Standard Deviation (Sigma)
c
      ErrorSum = 0
      Do 10401 point = 1,NumPts
      if (iwth(point) .eq. 0) go to 10401
      dvr(point) = spp(point,2) - u(jzW)*CosL(point)
     1            - v(jzW)*CosM(point) - w(jzW)*CosN(point)
      ErrorSum = ErrorSum + dvr(point)**2
10401 Continue
      Sigma = sqrt(ErrorSum/NPH)

      Do 10501 point = 1,NumPts
      if (iwth(point) .eq. 0) go to 10501
      if (abs(dvr(point)) .gt. NSigma*Sigma) then
      iwth(point) = 0
      flag = 1
      NPH = NPH - 1
      if (NPH .lt. MinH) then
      FitFlag = 0
      go to 90909
      endif
      endif
10501 Continue

      if (flag .eq. 0) go to 20002
      if (flag .eq. 1) then
      if (Sigma .ge. 0.999*SigmaLast) go to 20002
      if (Sigma .le. 0.01) go to 20002
      SigmaLast = Sigma
      go to 20001
      endif
c
c  good velocity.
c
20002 if ( (abs(u(jzW)) .gt. vHmax)      .or.
     1     (abs(v(jzW)) .gt. vHmax)      .or.
     2     (abs(w(jzW)) .gt. vHmax/20) ) then
      FitFlag = 0
      go to 90909
      endif

      WD = (180/pi)*atan2(v(jzW),u(jzW))

      do 10601 point = 1,NumPts
      if ((iwtv(point) .eq. 1) .or. (iwth(point) .eq. 1)) then
      Theta = (180/pi)*atan2(CosM(point),CosL(point))
      diff = Theta-WD
      if (diff .lt. -180) diff = diff + 360
      if (diff .gt. +180) diff = diff - 360

      if ((abs(diff) .lt. 45) .or. (abs(diff) .gt. 135)) then
      iDW = 1
```

```
      else
      iDW = 2
      endif
      PPCnt(iDW) = PPCnt(iDW) + 1
      PPCnt(3) = PPCnt(3) + 1

      dvr(point) = spp(point,2) - u(jzW)*CosL(point)
     1           - v(jzW)*CosM(point) - w(jzW)*CosN(point)
      ZA = (180/pi)*asin(sinZA(point))
      iZA = int(ZA) + 1

      if (iza .eq. 17) iZA = 16
      vrad(iZA,iDW) = vrad(iZA,iDW) + dvr(point)**2
      vrad(iZA,3) = vrad(iZA,3) + dvr(point)**2
c        vrad(iZA,iDW) = vrad(iZA,iDW) + abs(dvr(point))
c        vrad(iZA,3) = vrad(iZA,3) + abs(dvr(point))
      Numrad(iZA,iDW) = Numrad(iZA,iDW) + 1
      Numrad(iZA,3) = Numrad(iZA,3) + 1
      endif
10601 continue

      do 10702 iDir = 1,3
      if (PPCnt(iDir) .gt. 0) then
      do 10701 ialpha = 1,16
      if (Numrad(ialpha,iDir) .eq. 0) go to 10701
      vrad(ialpha,iDir) = sqrt(vrad(ialpha,iDir)/Numrad(ialpha,iDir))
c        vrad(ialpha,iDir) = vrad(ialpha,iDir)/Numrad(ialpha,iDir)
10701 continue
      endif
10702 continue

90909 Return
      End
```

```
 common /wind1/ Spp(5500,10)
 common /wind2/ z(130),u(130),v(130),w(130),TRP(130),
1             rej(7),line(10),sinZA(5500),Width(130),
2             iwtv(5500),iwth(5500),rmsdvr(130,3),PPCnt(3),
4             CosL(5500),CosM(5500),CosN(5500),dvr(5500),
5             Numrad(17,3),vrad(17,3),PPslope(2)
 common /wind3/ pi,vHmax,ThMaxH,ThMinH,ThMaxV,ThMinV,MinH,MinV,
1            Nsigma,Single,TestFlag,polarization,
1            jzW,QuitFlag,NumPts,interval,infile,outfile,
2            inpath,outpath,emonth,eday,ehour,eminute,NPH,NPV,NPV0,
3            slope,intercept,FitFlag
 real*4 pi,vHmax,ThMax,ThMaxV,dz,z,u,v,w,TRP,SigmaFinal,sinZA,
1       line,rmsdvr,CosL,CosM,CosN,dvr,slope,intercept,
2       vrad,PPslope
 integer*4 QuitFlag,rej,jzW,parameter,Single,TestFlag,
1          point,NumPts,interval,BigTime,NPV,NPV0,Testflag,FitFlag,
2          emonth,eday,ehour,eminute,MinH,MinV,
3          Numrad,PPCnt
 character*10 outpath
 character*40 infile,outfile
 character*9 inpath
 character*1 polarization
```

```
c
$Debug
      program Wind
c
**********************************************************************
*                                                                    *
*       IDI Wind-Calculation Program; MAPSTAR Radar.                 *
*       Copyright 1990, Holodyne Limited 1986.                       *
*       All Rights Reserved.                                         *
*                                                                    *
**********************************************************************
c
c   This program will calculate wind profiles in 3-km steps, with
c   smoothing, for a single scattering-point parameter file from SppM
c   or a group of files from SGroup.  The scattering-point parameters
c   are :
c   1.  Altitude (km).
c   2.  Radial velocity (m/sec).
c   3.  Zenith angle in East-West meridian (degrees).
c   4.  Zenith angle in North-South meridian (degrees).
c   5.  Voltage amplitude on #1 Dipoles.
c   6.  Phase of #1 Dipoles (degrees).
c   7.  Voltage amplitude on #2 Dipoles.
c   8.  Phase of #2 Dipoles (degrees).
c   9.  E-W zenith-angle window.
c   10. N-S zenith-angle window.
c
c   Explanation of easily-reprogrammed parameters (just change the source-
c      code value given below:
c   vHmax is the largest allowed horizontal velocity. We test each point
c     against Vmax by projecting its radial velocity into the horizontal
c     plane, and reject it if it's bigger than Vmax.
c   ThMaxV is the largest acceptable radial zenith angle for w.
c   ThMinV is the smallest acceptable radial zenith angle for w.
c   ThMaxH is the largest acceptable radial zenith angle for u and v.
c   ThMinH is the smallest acceptable radial zenith angle for u and v.
c   MinNumPts is the minimum number of points.  If there are not sufficient
c      points, that altitude is skipped.
c   NSigma is the maximum number of standard deviations from the fit any
c     individual point can lie without being rejected from the velocity
c     calculation.
c  Wind calls Header, inname, outname, WFV, WFH, and PhFit.
c
c  April 16, 1991.
c
$Include:'Wind.Inc'
$Include:'Header.inc'
      character*1 ans1,ans2,polarization
      real NumSndgs,Rate,Zmin,Zmax
      pi = 3.14159265
      polarization = 'a'
      VrMax = 300
      Zmin = 78
      Zmax = 102
      jzW = 26
      jzWmax = 34
c
c  Set ipick = 0/1 to disable/enable the vHmax filter.
c
      ipick = 1
```

```
      vHmax = +300

      ThMinV = 0
      ThMaxV = 10
      ThMinH = 3
      ThMaxH = 16
      ThMin = 0
      ThMax = 16

      MinH = 5
      MinV = 5
      Nsigma = 3.0

      CALL Header
      delZ = AltStep*le-3/2
************************************************************************
*   Communicate with User
*
20001 write (*,*) 'Single file, Group, or Loop? (s/g/L)'
      read (*,'(a)') ansl
      loop = 0
      if (ansl .eq. 'L') then
      ijk = 1
      loop = 1
      go to 20009
      elseif (ansl .eq. 's') then
      write (*,*) 'Input file name?'
      read (*,'(a)') infile
      outfile = 'Wind.dat'
      Single = 1
      elseif (ansl .eq. 'g') then
      Single = 0
      open (3,file='wind.txt')
      write (*,*) 'Use Default Values? (y/n)'
      read (*,'(a)') ans2
      if (ans2 .eq. 'y') then
c
c  user must set these values before compilation to use the default option.
c
      interval = 60
      inpath = 'e:\lhour\'
      outpath = 'c:\wlhour\'
      Month = 4
      Day = 5
      Hour = 14
      Minute = 00
      eMonth = 4
      eDay = 11
      eHour = 16
      eMinute = 0
c
cccccccccccccccccccccccccccccccccccccccccccccccccccc
      else
      write (*,*) 'Grouping interval (minutes)?'
      read (*,*) interval
      write (*,*) 'Center time of first file (month,day,hr,min)?'
      read (*,*) Month,Day,Hour,Minute
      write (*,*) 'Time of last file (month,day,hr,min)?'
      read (*,*) eMonth,eDay,eHour,eMinute
      write (*,*) 'Input path (9 characters)'
```

```
      read (*,'(a)') inpath
      write (*,*) 'Output path (10 characters)'
      read (*,'(a)') outpath
      endif
      else
      go to 20001
      endif
      go to 20010

20009 continue
      Month = 4
      Day = 5
      Hour = 14
      Minute = 00
      eMonth = 4
      eDay = 11
      eHour = 16
      if (ijk .eq. 1) then
      interval = 5
      inpath = 'e:\05min\'
      outpath = 'c:\w05min\'
      elseif (ijk .eq. 2) then
      interval = 15
      inpath = 'e:\15min\'
      outpath = 'c:\w15min\'
      elseif (ijk .eq. 3) then
      interval = 60
      inpath = 'e:\1hour\'
      outpath = 'c:\w1hour\'
      elseif (ijk .eq. 4) then
      interval = 120
      inpath = 'e:\2hour\'
      outpath = 'c:\w2hour\'
      endif
20010 continue
************************************************************************

      if (Single .eq. 1) go to 20102
      open (3,file='Wind.txt')
************************************************************************
*    Return to here for new file
************************************************************************
20101 call inname
      call outname
      write (*,*) 'infile = ',infile
      write (*,*) 'outfile = ',outfile
      write (3,'(a)') outfile

20102 write (*,90003)
90003 format
     1 (1x,' alt    u       v     w   TRP  Ntot  Rate',
     2 '  slope intercept PPs(1)  PPs(2)')
      Open (1,err=90909,file=infile,status='old',form='binary')
20103 Read (1,err=90909,end=90909) (line(parameter),parameter=1,10)
      if (line(1) .gt. -990) go to 20103
      NumSndgs = line(2)
      rewind (1)
      Open (2,err=90909,file=outfile)
      QuitFlag = 0
c     jzW = 1
```

```
      do 10101 jz = 1,40
      Z(jz) = (AltStep*float(jz-1) + AltMin)*1e-3
      u(jz) = 0
      v(jz) = 0
      w(jz) = 0
      TRP(jz) = 0
      rmsdvr(jz,1) = 0
      rmsdvr(jz,2) = 0
10101 continue

      do 10102 irej=1,4
      rej(irej) = 0
10102 continue
c     Read (1,err=90909,end=20203) (line(parameter),parameter=1,10)

************************************************************************
*  Return to here for new altitude.
************************************************************************
c
20201 NumPts = 0
      rewind (1)
      Read (1,err=90909,end=20203) (line(parameter),parameter=1,10)

20202 if (line(1)  .le. Z(jzW)-delZ) then
      Read (1,err=90909,end=20203) (line(parameter),parameter=1,10)
      go to 20202
      endif
      if (line(1) .gt. Z(jzW)+delZ) then
      if ((NumPts .lt. MinV) .or. (NumPts .lt. MinH)) go to 20206
      go to 20204
      endif
c
c  Reject the point if: (1) altitude < 1 km.
c                       (2) zenith angle not between ThMin and ThMax,
c                       (3) projected horizontal velocity > vHmax,
c                       (4) radial velocity = 0
c                       (5) linear polarization
c                       (6) ordinary polarization
c                       (7) extraordinary polarization
c
      TestFlag = 1

c     if ((line(3) .gt. 0) .and.
c    1    (line(4) .lt. 0)) then
      if (line(1) .lt. 1) then
      rej(1) = rej(1) + 1
      TestFlag = 0
      endif

      sinZAx = sqrt(sin(line(3)*pi/180)**2
     1           + sin(line(4)*pi/180)**2)
      if ( (sinZAx .lt. sin(ThMin*pi/180))   .or.
     1     (sinZAx .gt. sin(ThMax*pi/180)) ) then
      rej(2) = rej(2) + 1
      TestFlag = 0
      endif

      if (sinZAx .gt. 0.02 .and. ipick .eq. 1) then
      if(abs(line(2)/sinZAx) .gt. vHmax) then
      rej(3) = rej(3) + 1
```

```
      TestFlag = 0
      endif
      endif

      if (line(2) .eq. 0) then
      rej(4) = rej(4) + 1
      TestFlag = 0
      endif

      if (abs(line(2)) .gt. VrMax) then
      TestFlag = 0
      endif

c
c  Linear polarization filter (removes linearly polarized points):
c
      if (polarization .eq. 'o' .or. polarization .eq. 'x') then
      if (abs(line(6)-line(8)) .lt. 45)  TestFlag = 0
      if ( (abs(line(6)-line(8)) .gt. 135) .and.
     1       (abs(line(6)-line(8)) .lt. 225) ) TestFlag = 0
      if ( (abs(line(6)-line(8)) .gt. 315) .and.
     1       (abs(line(6)-line(8)) .lt. 360) ) TestFlag = 0
      endif
c
c
c  Ordinary polarization filter (removes ordinary points):
c
      if (polarization .eq. 'l' .or. polarization .eq. 'x') then
      if (line(6)-line(8) .gt. 45  .and.
     1    line(6)-line(8) .lt. 135 )  TestFlag = 0
      if (line(6)-line(8) .gt. -315  .and.
     1    line(6)-line(8) .lt. -225 )  TestFlag = 0
      endif
c
c
c  Extraordinary polarization filter (removes extraordinary points):
c
      if (polarization .eq. 'l' .or. polarization .eq. 'o') then
      if (line(6)-line(8) .gt. -135  .and.
     1    line(6)-line(8) .lt. -45 )  TestFlag = 0
      if (line(6)-line(8) .gt. 225  .and.
     1    line(6)-line(8) .lt. 315 )  TestFlag = 0
      endif
c
c
c
**********************************************************************
      if (NumPts .eq. 6500) then
      write (*,*) 'Thanks anyhow, but Ive already got 6500 points.'
      TestFlag = 0
      endif

c     else
c     TestFlag = 0
c     endif

      if (TestFlag .eq. 1) then
      NumPts = NumPts + 1

      do 10201 parameter = 1,10
```

```
      spp(NumPts,parameter) = line(parameter)
10201 continue

      TRP(jzW) = TRP(jzW) + line(5)**2 + line(7)**2
      endif

      Read (1,err=90909,end=20203) (line(parameter),parameter=1,10)
      go to 20202

20203 quitflag = 1

20204 Fitflag = 1
c
c  Fit the scattering points in this window with a 3-vector.
c
20205 CALL WFV
      if (Fitflag .eq. 0) then
      write (*,*) 'Vertical Failure at ',jzW,Z(jzW)
      write (2,90002) -1000.0,-1000.0,-1000.0,-1000.0,-1000.0,-1000.0,
     1                -1000.0,-1000.0,-1000.0,-1000.0,-1000.0,-1000.0
      go to 20206
      endif

      Call WFH

      if (Fitflag .eq. 0) then
      write (*,*) 'Horizontal Failure at ',jzW,Z(jzW)
      write (2,90002) -1000.0,-1000.0,-1000.0,-1000.0,-1000.0,-1000.0,
     1                -1000.0,-1000.0,-1000.0,-1000.0,-1000.0,-1000.0
      else
      if (TRP(jzW) .lt. 1) then
      TRP(jzW) = 0
      else
      TRP(jzW) = 10*log10(TRP(jzW))
      endif
      call PhFit
      Rate = float(NumPts)/NumSndgs
      write (*,90001)
     1 Z(jzW),u(jzW),v(jzW),w(jzW),TRP(jzW),NumPts,Rate,
     2 slope,intercept,PPslope(1),PPslope(2)
      x1 = float(NumPts)
      x2 = float(NPV)
      x3 = float(NPH)
      write (2,90002)
     1  Z(jzW),u(jzW),v(jzW),w(jzW),w(jzW)*10,TRP(jzW),x1,Rate,
     2  slope,intercept,PPslope(1),PPslope(2)
      endif
90001 format (1x,f4.0,2(1x,f6.1),2(1x,f5.1),1x,i4,5(1x,f5.2))
90002 format (1x,12(e12.4,1x))
c
c  If it's not time to quit, increment jzW and go read the next points.
c
20206 if (QuitFlag .eq. 0) then
      jzW = jzW + 1
      if (jzW .le. jzWmax) then
c     Read (1,err=90909,end=20203) (line(parameter),parameter=1,10)
      go to 20201
      endif
      endif
```

```
      Close (1)
      Close (2)

      write (*,*) 'Rejections:'
      WRITE (*,*) '        Z<1km      ThLimits       vHmax        Vr=0'
      write (*,*) (rej(irej),irej=1,4)

      If (Single .eq. 1) go to 90909
      if (month .eq. emonth .and. day .eq. eday .and.
     1    hour .eq. ehour .and. minute .eq. eminute) go to 90909
      BigTime = minute+hour*60+day*24*60+month*30*24*60 + interval
      month = BigTime/(30*24*60)
      day = (Bigtime-month*30*24*60)/(24*60)
      hour = (BigTime-month*30*24*60-Day*24*60)/60
      minute = BigTime-month*30*24*60-Day*24*60-Hour*60
      go to 20101
90909 close (1)
      close (2)
      close (3)
91919 if (loop .eq. 1) then
      ijk = ijk + 1
      if (ijk .le. 4) go to 20009
      endif

      End
```

```
c
$Debug
c
      Subroutine inName
c
c  inName creates the input file names for WGroup.
c
$Include:'Wind.inc'
$Include:'Header.inc'
c
c
      character*2 ascmonth,ascday,aschour,ascminute
      if (month .lt. 10) then
      write (ascmonth,90001) '0',month
90001 format (a1,i1)
      else
      write (ascmonth,90002) month
90002 format (i2)
      endif
      if (day .lt. 10) then
      write (ascday,90001) '0',day
      else
      write (ascday,90002) day
      endif
      if (hour .lt. 10) then
      write (aschour,90001) '0',hour
      else
      write (aschour,90002) hour
      endif
      if (minute .lt. 10) then
      write (ascminute,90001) '0',minute
      else
      write (ascminute,90002) minute
      endif
      write (infile,90003)
     1  inpath,ascMonth,ascDay,ascHour,ascMinute,'.mbs'
90003 format (21a)
      return
      end
```

```
c
$Debug
c
      Subroutine outName
c
c  outName creates the output file names for WGroup.
c
$Include:'Wind.inc'
$Include:'Header.inc'
c
c
      character*2 ascmonth,ascday,aschour,ascminute
      if (month .lt. 10) then
      write (ascmonth,90001) '0',month
90001 format (a1,i1)
      else
      write (ascmonth,90002) month
90002 format (i2)
      endif
      if (day .lt. 10) then
      write (ascday,90001) '0',day
      else
      write (ascday,90002) day
      endif
      if (hour .lt. 10) then
      write (aschour,90001) '0',hour
      else
      write (aschour,90002) hour
      endif
      if (minute .lt. 10) then
      write (ascminute,90001) '0',minute
      else
      write (ascminute,90002) minute
      endif
      write (outfile,90003)
     1  outpath,ascMonth,ascDay,ascHour,ascMinute,'.maw'
90003 format (21a)
      return
      end
```

```
c
$Debug
c
*********************************************************************
      Subroutine PhFit
*********************************************************************
C
C   THIS SUBROUTINE fits two straight lines to the variation of velocity
c   variance vs zenith angle; one line to variations along the wind
c   vector, the second to variations perpendicular to the wind vector.
C   August 18, 1990
c
$Include:'Wind.Inc'
$Include:'Header.inc'
c
c  First, use all the points to get the intercept.
c
      sumvr = 0
      sumvrph = 0
      sumph = 0
      sumph2 = 0
      sumI = 0


      do 10101 ialpha = 1,17
      if (Numrad(ialpha,3) .eq. 0) go to 10101
      ZA = ialpha - 0.5
      sumvr = sumvr + vrad(ialpha,3)
      sumvrph = sumvrph + vrad(ialpha,3)*ZA
      sumph = sumph + ZA
      sumph2 = sumph2 + ZA**2
      sumI = sumI + 1
10101 continue
10102 continue

      if ((sumI .ge. 3) .and.
     1    (sumI*sumph2 - sumph**2 .ne. 0)) then
      slope = (sumI*sumvrph - sumvr*sumph)/(sumI*sumph2 - sumph**2)
      intercept = (sumvr - slope*sumph)/sumI
      else
      slope = 0
      intercept = 0
      PPslope(1) = 0
      PPslope(2) = 0
      return
      endif
c
c  Now fit the parallel and perpendicular variances separately.
c
      do 10202 iDir = 1,2
      sumvr = 0
      sumvrph = 0
      sumph = 0
      sumph2 = 0
      sumI = 0

      do 10201 ialpha = 1,17
      if (Numrad(ialpha,iDir) .eq. 0) go to 10201
      ZA = ialpha - 0.5
      sumvr = sumvr + vrad(ialpha,iDir)
```

```
      sumph = sumph + ZA
      sumI = sumI + 1
10201 continue
      PPslope(iDir) = 0
      if (sumI .gt. 0) PPslope(iDir)=(sumvr-intercept*sumI)/sumph
10202 continue

c     write (*,*) 'NPH,PPCnt(1),PPCnt(2),PPCnt(3) = '
c     write (*,*)  NPH,PPCnt(1),PPCnt(2),PPCnt(3)
c     do 10301 ialpha = 1,17
c     write (*,*)
c    1 ialpha,NumRad(ialpha,1),NumRad(ialpha,2),NumRad(ialpha,3)
10301 continue
      return
      end
```

```
c
$Debug
c
      program SGroup
c
c  SGroup (Scattering-point Grouping software) takes a list of (presumably
c  tape-long) scattering-point parameter (.mbs) files (which are specified
c  in SGroup.txt), groups them into
c  time intervals specified by the user, sorts them by altitude, and
c  names them according to the time at the center of the interval.
c  Filtering by zenith angle (ThetaMax) and maximum projected horizontal
c  velocity (Vmax) are also done here.
c  Converted to 10 parameters/point: 3/21/91
c
$Include:'SGroup.inc'
$include:'sname.inc'
c
c  Link Sgroup+SName+SMerge+BellSub
c
***********  SET-UP AND INITIALIZE  ************
c
      real*4 xnumsndgs
      integer*4 filecount
      character*3 filter
      character*1 answer
      pi = 3.14159265
      ThetaMax = 16
      vHmax = 300
      Zmin = 60
      Zmax = 120
      filter = 'on'
      write (*,*) 'SGroup expects the list of input (.mbs) files and'
      write (*,*)
     1'their polarization codes to be in SGroup.txt.'
      path = 'd:\1hour\'
      write (*,*) 'Single file? (y/n)'
      read (*,'(a)') answer
      if (answer .eq. 'y') then
      write (*,*) 'DataSpan,month,day,hour,min = '
      read (*,*) DataSpan,month,day,hour,minute
      polarization(1) = 'o'
      else
      DataSpan = 60
      Spacing = 60
      month = 4
      day = 5
      hour = 14
      minute = 00
      endif
      call SName
      write (*,*) 'Ready to Fill First Output File ',sfile
      open (102,file=sfile,form='binary')
      filecount = 1
      BigTime = Minute + Hour*60 + Day*24*60 + Month*30*24*60
      FirstTime = BigTime - DataSpan/2
      LastTime = BigTime + DataSpan/2
      open (101,file='SGroup.txt',status='old')
      iTape = 1
20001 read (101,90201,end=20002) infile(iTape),polarization(iTape)
90201 format (a,2x,a)
```

```
      iTape = iTape + 1
      go to 20001
20002 close (101)
      NumTapes = iTape-1
      write (*,*) 'Number of tapes to process = ',NumTapes
      do 10001 iSndg = 1,9
      write (asciSndg,90101) '000',iSndg
90101 format (a3,i1)
      write (Sndg(iSndg),90005) 'c:\holda\',asciSndg,'.mbs'
10001 continue
90005 format (17a)
      do 10002 iSndg = 10,99
      write (asciSndg,90102) '00',iSndg
90102 format (a2,i2)
      write (Sndg(iSndg),90005) 'c:\holda\',asciSndg,'.mbs'
10002 continue
      do 10003 iSndg = 100,500
      write (asciSndg,90103) '0',iSndg
90103 format (a1,i3)
      write (Sndg(iSndg),90005) 'c:\holda\',asciSndg,'.mbs'
10003 continue
      do 10004 iSndg = 501,999
      write (asciSndg,90103) '0',iSndg
      write (Sndg(iSndg),90005) 'c:\holdb\',asciSndg,'.mbs'
10004 continue
      iTape = 1
      Now = 0
      iSndg = 0
******************  NOW START PROCESSING TAPES  ****************
c
c  Return to here when another input tape is needed.
c
20003 write (*,*) 'Ready to Process Tape ',infile(iTape)
      write (*,*) 'Polarization: ',polarization(iTape)
      open (101,file=infile(iTape),form='binary')
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
c
c  Return to here when another output file is needed.
c
20004 iSndg = 0
      Now = 0
      filecount = filecount + 1
      call SName
      write (*,*) 'Ready to fill Output File ',sfile
      open (102,file=sfile,form='binary')
20005 if (spp(1) .lt. -990) then
      iswitch = 0
      ThisTime = spp(3)*30*24*60 + spp(4)*24*60 + spp(5)*60 + spp(6)
      if (ThisTime .lt. FirstTime) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      elseif (ThisTime .gt. LastTime) then
      go to 20007
      else
      Now = 1
      Select = 0
      endif
      endif
c
```

```
c  Go back; not time yet.
c
      if (Now .eq. 0) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
      if (filter .eq. 'off') go to 20006
c
c  Filter by altitudes.
c
      if (spp(1) .lt. Zmin .or. spp(1) .gt. Zmax) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
c
c  Polarization Filters:
c   o = ordinary
c   p = ordinary plus linear
c   x = extraordinary
c   y = extraordinary plus linear
c   l = linear
c   c = circular = ordinary plus extraordinary
c   n = none (filtering done at 20009).
c
c  linear polarization filter
c
      if ((polarization(iTape) .eq. 'o') .or.
     1    (polarization(iTape) .eq. 'x') .or.
     2    (polarization(iTape) .eq. 'c')) then
      pd = abs(spp(6)-spp(8))
      if ( (pd .le. 45) .or.
     1   ( (pd .ge. 135) .and. (pd .le. 225) ) .or.
     2     (pd .ge. 315) ) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
      endif
c
c  ordinary polarization filter
c
      if ((polarization(iTape) .eq. 'x') .or.
     1    (polarization(iTape) .eq. 'y') .or.
     2    (polarization(iTape) .eq. 'l')) then
      if (spp(6)-spp(8) .gt. 45 .and. spp(6)-spp(8) .lt. 135) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
      if (spp(6)-spp(8) .gt. -315 .and. spp(6)-spp(8) .lt. -225) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
      endif
c
c  extraordinary polarization filter:
c
      if ((polarization(iTape) .eq. 'o') .or.
     1    (polarization(iTape) .eq. 'p') .or.
     2    (polarization(iTape) .eq. 'l')) then
      if (spp(6)-spp(8) .ge. -135 .and.
     1    spp(6)-spp(8) .le. -45) then
```

```
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
      if (spp(6)-spp(8) .ge. 225 .and.
     1    spp(6)-spp(8) .le. 315) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
      endif
c
c  Filter on zenith angle.
c
      sinZA = sqrt(sin(spp(3)*pi/180)**2 + sin(spp(4)*pi/180)**2)
      if (asin(sinZA)*180/pi .gt. ThetaMax) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
c
c  Filter on vHmax.
c
      if (sinZA .gt. .01) then
      if (abs(spp(2)/sinZA) .gt. vHmax) then
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
      endif
      endif

20006 if (iswitch .eq. 0) then
      iswitch = 1
      iSndg = iSndg + 1
      if (iSndg .gt. 999) go to 20008
      close (103)
      open (103,file=Sndg(iSndg),form='binary')
      endif

      if (spp(1) .gt. -990)
     1    write (103) (spp(parameter),parameter=1,10)
      read (101,end=20008) (spp(parameter),parameter=1,10)
      go to 20005
c
c  if you're past the time limit, merge the individual output files into
c  the outfile, set the new time limits, and go do the next output file.
c
20007 close  (103)
      NumSndgs = iSndg
      write (*,*) 'Number of Soundings in this group:',NumSndgs
      write (*,*) ' '
      if (NumSndgs .gt. 0) call SMerge
      xnumsndgs = float(numsndgs)
      write (102) -999.0, xnumsndgs, .0, .0, .0, .0, .0, .0, .0, .0
      close (102)
      if (answer .eq. 'y') go to 90909
      if (iTape .eq. NumTapes+1) go to 90909
      BigTime = BigTime + Spacing
      Month = BigTime/(30*24*60)
      Day = (BigTime-Month*30*24*60)/(24*60)
      Hour = (BigTime-Month*30*24*60-Day*24*60)/60
      Minute = (BigTime-Month*30*24*60-Day*24*60-Hour*60)
      FirstTime = BigTime - DataSpan/2
      LastTime = BigTime + DataSpan/2
```

```
      go to 20004

20008 close (101)
      iTape = iTape + 1
20009 if (polarization(iTape) .eq. 'n') then
      iTape = iTape + 1
      go to 20009
      endif
      if (iTape .le. NumTapes) go to 20003
      if (iTape .eq. NumTapes+1) go to 20007
90909 close (201)
      call BellSub
      end
```

```
c
$Debug
c
      Subroutine SName
c
c  SName creates the file names for SGroup.
c
$Include:'Sname.inc'
c
c
      if (month .lt. 10) then
      write (ascmonth,90001) '0',month
90001 format (a1,i1)
      else
      write (ascmonth,90002) month
90002 format (i2)
      endif
      if (day .lt. 10) then
      write (ascday,90001) '0',day
      else
      write (ascday,90002) day
      endif
      if (hour .lt. 10) then
      write (aschour,90001) '0',hour
      else
      write (aschour,90002) hour
      endif
      if (minute .lt. 10) then
      write (ascminute,90001) '0',minute
      else
      write (ascminute,90002) minute
      endif
      write (sfile,90003)
     1  path,ascMonth,ascDay,ascHour,ascMinute,'.mbs'
90003 format (28a)
      return
      end
```

```
$Debug
c
      Subroutine SMerge
c
c  This subroutine will merge a number of binary scattering-point
c  parameter (.mbs) files generated by SGroup into a single file sorted
c  by altitude.  The names of the files are c:\holda\0001.mbs to 0500.mbs,
c  and c:\holdb\0501.mbs to 0999.mbs.
c  December 30, 1989.
c  Modified for 10-parameter points: 3/21/91
c  Looking for problem that results in altitude order problem: 5/30/91.
c
$Include:'SGroup.inc'
      dimension spp1(10),spp2(10)
      real*4 zlast
      integer*4 index
*************************************************************************
c
c  read the first file into temp1.mbs to get started.
c
      open (1,file=Sndg(1),status='old',form='binary')
      open (2,file='c:temp1.mbs',status='unknown',form='binary')
20001 read (1,end=20002) (spp1(parameter),parameter=1,10)
      write (2) (spp1(parameter),parameter=1,10)

      go to 20001
20002 imark = 1
      close (1)
      close (2)

      if (NumSndgs .eq. 2) go to 20030
      if (NumSndgs .eq. 1) go to 20040

*************************************************************************
*********************** Process Files By Pairs ************************
c
c         ************* First File of Pair **************
c
      do 10002 ifile = 2,NumSndgs-1,2

      open (1,file=Sndg(ifile),status='old',form='binary')
      open (2,file='c:temp1.mbs',status='old',form='binary')
      open (3,file='c:temp2.mbs',status='unknown',form='binary')
40001 read (1,end=20013) (spp1(parameter),parameter=1,10)
40002 read (2,end=20012) (spp2(parameter),parameter=1,10)
20011 if (spp1(1) .lt. spp2(1)) then
      write (3) (spp1(parameter),parameter=1,10)

40003 read (1,end=20013) (spp1(parameter),parameter=1,10)
      go to 20011

      else
      write (3) (spp2(parameter),parameter=1,10)

40004 read (2,end=20012) (spp2(parameter),parameter=1,10)
      go to 20011
      endif
c
c  you get here if temp1.vbs ran out of points before Sndg.
```

```
c
20012 write (3) (spp1(parameter),parameter=1,10)

20112 read (1,end=20014) (spp1(parameter),parameter=1,10)
      write (3) (spp1(parameter),parameter=1,10)

      go to 20112
c
c  you get here if Sndg ran out of points before templ.vbs.
c
20013 write (3) (spp2(parameter),parameter=1,10)

20113 read (2,end=20014) (spp2(parameter),parameter=1,10)
      write (3) (spp2(parameter),parameter=1,10)

      go to 20113

20014 imark = imark + 1
      close (1)
      close (2)
      close (3)
c
c          ************** Second File of Pair **************
c
      open (1,file=Sndg(ifile+1),status='old',form='binary')
      open (2,file='c:temp2.mbs',status='unknown',form='binary')
      open (3,file='c:templ.mbs',status='unknown',form='binary')

40005 read (1,end=20023) (spp1(parameter),parameter=1,10)
40006 read (2,end=20022) (spp2(parameter),parameter=1,10)
20021 if (spp1(1) .lt. spp2(1)) then
      write (3) (spp1(parameter),parameter=1,10)

40007 read (1,end=20023) (spp1(parameter),parameter=1,10)
      go to 20021
      else
      write (3) (spp2(parameter),parameter=1,10)

40008 read (2,end=20022) (spp2(parameter),parameter=1,10)
      go to 20021
      endif
c
c  you get here if templ.vbs ran out of points before Sndg.
c
20022 write (3) (spp1(parameter),parameter=1,10)

20122 read (1,end=20024) (spp1(parameter),parameter=1,10)
      write (3) (spp1(parameter),parameter=1,10)

      go to 20122
c
c  you get here if Sndg ran out of points before templ.vbs.
c
20023 write (3) (spp2(parameter),parameter=1,10)

20123 read (2,end=20024) (spp2(parameter),parameter=1,10)
      write (3) (spp2(parameter),parameter=1,10)

      go to 20123
```

```
20024 imark - imark + 1
      close (1)
      close (2)
      close (3)

10002 continue
      if (NumSndgs-imark .eq. 1) go to 20030
      if (NumSndgs-imark .eq. 0) go to 20040

************************************************************************
c
c  if the number of files is even, there will still be one file left.
c

20030 open (1,file-Sndg(NumSndgs),status-'old',form-'binary')
      open (2,file-'c:templ.mbs',status-'old',form-'binary')
40009 read (1,end-20033) (sppl(parameter),parameter-1,10)
40010 read (2,end-20032) (spp2(parameter),parameter-1,10)
20031 if (sppl(1) .lt. spp2(1)) then
       write (102) (sppl(parameter),parameter-1,10)

40011 read (1,end-20033) (sppl(parameter),parameter-1,10)
      go to 20031

      else
      write (102) (spp2(parameter),parameter-1,10)

40012 read (2,end-20032) (spp2(parameter),parameter-1,10)
      go to 20031
      endif
c
c  you get here if templ.vbs ran out of points before Sndg.
c
20032 write (102) (sppl(parameter),parameter-1,10)

20132 read (1,end-30001) (sppl(parameter),parameter-1,10)
      write (102) (sppl(parameter),parameter-1,10)

      go to 20132
c
c  you get here if Sndg ran out of points before templ.vbs.
c
20033 write (102) (spp2(parameter),parameter-1,10)

20133 read (2,end-30001) (spp2(parameter),parameter-1,10)
      write (102) (spp2(parameter),parameter-1,10)

      go to 20133

************************************************************************
c
c  If the number of files was odd, or there was only one file
c  initially, transfer the sorted file into mergeout.vbs.
c
20040 open (1,file-'c:templ.mbs',status-'old',form-'binary')
20041 read (1,end-30001) (sppl(parameter),parameter-1,10)
      write (102) (sppl(parameter),parameter-1,10)

      go to 20041
```

```
*************************************************************************
30001 close (1)
      close (2)
      close (3)
      end
```

```
c
$Debug
c
      subroutine Bellsub
      integer*2 ihr,imin,isec,i100th
      integer*4 duration
      real*4 LastTime,ThisTime
      character*1 ding
      Call gettim(ihr,imin,isec,i100th)
      Duration = 0
      LastTime = ihr*3600 + imin*60 + isec + i100th*.01
      ding = char(7)

20001 write (*,90001) ding
90001 format (a1,\)
20002 Call gettim(ihr,imin,isec,i100th)
      ThisTime = ihr*3600 + imin*60 + isec + i100th*.01
      if (ThisTime - LastTime .lt. 1) go to 20002
      Last Time = ThisTime
      Duration = Duration + 1
      if (Duration .le. 10) go to 20001
      return
      end
```

# sname.inc

```
c sname.inc
      common /sname1/ month,day,hour,minute,sfile,path
      character*2 ascmonth,ascday,aschour,ascminute
      integer*4 month,day,hour,minute
      character*9 path
      character*40 sfile
```

# SGroup.inc

```
c
c
c  SGroup.inc
c
      common /SG1/ BigTime,pi,ThetaMax,Vmax,NumSndgs,Sndg
c     common /SG2/ SumNum(44),SumPwr(44)
      dimension spp(10),infile(1000),Sndg(1000),polarization(1000)
      real*8 SumNum,SumPwr
      real*4 spp,ThetaMax,Vmax,pi
      integer*4 iTape,group,NoiseLimit,NoiseCount,iSndg,
     1          BigTime,FirstTime,LastTime,DataSpan,Spacing,
     2          NumSndgs,parameter,select,Now
      character*4 ascisndg
      character*16 infile
      character*17 Sndg
      character*1 polarization
```

```
DatLog.txt

Dat #01
GR233: 50 raw files; list, 49 processed files, .mbs file. 101   101
GR235: 50 raw files; list, 49 processed files, .mbs file. 101   202
GR236: 50 raw files; list, 49 processed files, .mbs file. 101   303
GR239: 49 raw files; list, 47 processed files, .mbs file.  98   401
GR240: 49 raw files; list, 47 processed files, .mbs file.  98   499
GR241: 48 raw files; list, 45 processed files, .mbs file.  95   594
GR242: 50 raw files; list, 49 processed files, .mbs file. 101   695
GR243: 50 raw files; list, 49 processed files, .mbs file. 101   796
GR244: 49 raw files; list, 47 processed files, .mbs file.  98   894
GR245: 50 raw files; list, 49 processed files, .mbs file. 101   995

Dat #02
GR246: 50 raw files; list, 49 processed files, .mbs file. 101   101
GR247: 46 raw files; list, 43 processed files, .mbs file.  91   192
GR248: 50 raw files; list, 49 processed files, .mbs file. 101   293
GR249: 50 raw files; list, 49 processed files, .mbs file. 101   394
GR250: 50 raw files; list, 49 processed files, .mbs file. 101   495
GR251: 50 raw files; list, 49 processed files, .mbs file. 101   596
GR252: 50 raw files; list, 49 processed files, .mbs file. 101   697
GR253: 49 raw files; list, 45 processed files, .mbs file.  96   793
GR254: 50 raw files; list, 49 processed files, .mbs file. 101   894
GR255: 49 raw files, list, 31 processed files, .mbs file.  82   976

Dat #03 (Djuth)
GR267: 40 raw files, list, 38 processed files, .mbs file.   80    80
GR268: 41 raw files, list, 38 processed files, .mbs file.   80   160
GR292: 41 raw files, list, 39 processed files, .mbs file.   82   242
GR293: 40 raw files, list, 38 processed files, .mbs file.   80   322
GR334: 49 raw files, list, 48 processed files, .mbs file.   99   421
GR335: 49 raw files, list, 47 processed files, .mbs file.   98   519
GR336: 49 raw files, list, 48 processed files, .mbs file.   99   618

Dat #04
GR160: 49 raw files, list, 48 processed files, .mbs file.   99    99
GR161: 49 raw files, list, 48 processed files, .mbs file.   99   198
GR162: 49 raw files, list, 48 processed files, .mbs file.   99   297
GR163: 48 raw files, list, 47 processed files, .mbs file.   97   394
GR164: 49 raw files, list, 48 processed files, .mbs file.   99   493
GR165: 48 raw files, list, 46 processed files, .mbs file.   96   589
GR166: 47 raw files, list, 44 processed files, .mbs file.   93   682
GR167: 49 raw files, list, 48 processed files, .mbs file.   99   781
GR168: 49 raw files, list, 48 processed files, .mbs file.   99   880
GR169: 49 raw files, list, 48 processed files, .mbs file.   99   979

Dat #05
GR170: 49 raw files, list, 48 processed files, .mbs file.   99    99
GR171:    raw files, list,    processed files, .mbs file.
GR172:    raw files, list,    processed files, .mbs file.
GR173:    raw files, list,    processed files, .mbs file.
```

## The DAT to disc copying program TAPEREAD.f

This, and the following Macintosh II compatible FORTRAN 77 programs have been developed by Bob Roper to facilitate the comparisons of IDI - ISR and IDI - Fabry-Perot spectrometer (FPS) winds, with appropriately acknowledged portions of the Adams' programs used to obviate reinvention.

TAPEREAD.f copies the scattering point parameter files from DAT to hard drive as SPP - GR XXX files (where XXX is the original MAPSTAR 9 track data tape number) for subsequent processing by programs IDIWIND.f, GROVES.f and ISRIDIIDIG.f. This may seem redundant, but the original scattering point parameter files contain Universal Coordinated Time (UTC) data from close to the surface to as high as 130km.TAPEREAD changes the timing to local mean solar time (LMST = UTC - 4 hours 28 minutes for Arecibo, which is at 18°N, 67°W. The change to LMST conforms to the international protocol for the reporting of atmospheric tidal wind phases) and records only those scattering points whose altitudes are between 66 and 116km, which spans the altitude range (70 - 95Km) of primary interest in the AIDA comparisons; the altitudes from 66 to 116km are used in the IDI wind and GROVES analyses. A considerable amount of time in subsequent processing is saved because of not only the smaller data set, but also the decreased data access time..

TAPEREAD uses a subroutine IQTAPE, which is a proprietary item. The WangDAT tape drive accessing IQTAPE routines may be purchased from

Cyber-Comp Inc.,<br>10522 Topeka Drive,<br>Northridge, CA 91326-3032

Phone (818) 366-6786

```
C     PROGRAM TAPEREAD
C     READS WangDAT TAPE, WRITES SPP - GR XXX FILES TO DISC
C     (ONE FILE PER RAW DATA TAPE)
C     AND PRINTS STATISTICS TO SCREEN.
C     REJECTS ALL DATA OUTSIDE HEIGHT RANGE ZMIN TO ZMAX,
C     AND ALL ABS(ARRIVAL ANGLES) > 90 DEGREES.
C     CONVERTS TIMES TO LOCAL MEAN SOLAR (ARECIBO; 18N,67W)
C
C     REQUIRES SUBROUTINES
C                         RDYWANG
C                         SWITCH7 (IBM TO MAC REALS)
C                         IBAD
C                         IQTAPE
C
      REAL*4 SPBLOCK(10),SPBYTE(10),FLAG
      INTEGER*1 STATS(64)
      INTEGER*4 CHAN,IFIX,NUM,ISIZE,NBOUT,SKIP
      CHARACTER*1 DUM(40),BITE(16384)
      CHARACTER*40 DUMMY
      COMMON STATS,BITE
      EQUIVALENCE (DUMMY,DUM)
      LAST=341
      ZMAX=116.0
      ZMIN=66.0
      FLAG=-999.0
      NOUGHT=0
      ZERO=0.0
      NBAD=NOUGHT
      NINETY=90.0
      CHAN=3
      IFIX=NOUGHT
      NUM=1
      SIZE=16384
      NBOUT=16384
      HOLD=ZERO
C
      WRITE (*,*) " PROGRAM TAPECOPY - WangDAT TO DISC"
      WRITE (*,*) " "
C
      WRITE (*,*) " ENTER FIRST SPP TAPE NUMBER THIS DAT TAPE"
      READ (*,*) ITAPE
      WRITE (*,*) " ENTER START TAPE NUMBER"
      READ (*,*) NTAPE
      WRITE (*,*) " ENTER LAST TAPE NUMBER"
      READ (*,*) LAST
      SKIP=NTAPE-ITAPE
      CALL RDYWANG (SKIP)
      OPEN (17,FILE="DIAGNOS",FORM="FORMATTED")
C
    6 LOOP=NOUGHT
      NGOOD=NOUGHT
      WRITE (*,*) " PROCESSING TAPE ",NTAPE
      WRITE (17,*) " PROCESSING TAPE ",NTAPE
C
C     DEFINE SCRATCH TAPE
C
```

```
      OPEN (16,FILE="TEMP",FORM="UNFORMATTED")
C
C     OPEN DISC FILE FOR NTAPE
C
      DUMMY="SPP - GR "
      I100=NTAPE/100
      I10=(NTAPE-I100*100)/10
      I1=NTAPE-I100*100-I10*10
      DUM(10)=CHAR(I100+48)
      DUM(11)=CHAR(I10+48)
      DUM(12)=CHAR(I1+48)
      OPEN (26,FILE=DUMMY,FORM="UNFORMATTED")
C
    7 JERR=IQTAPE (15,CHAN,IFIX,NUM,BITE,SIZE,NBOUT,STATS)     !READ TAPE
      IF (JERR.EQ.8) GO TO 20
      IF (JERR.GT.-1) GO TO 15
      IF (JERR.EQ.-1) GO TO 16
      WRITE (*,*) "+++++ TAPE READ ERROR +++++ JERR = ",JERR,HOLD
      GO TO 7
C
C     WRITE TEMPORARY FILE
C
   15 WRITE(16) BITE
      GO TO 7
C
C     SELECT USEFUL DATA FROM FILE "TEMP"
C
   16 REWIND (16)
   30 READ (16,END=33) SPBYTE
      CALL SWITCH7 (SPBLOCK,SPBYTE)
      IF (SPBLOCK(1).EQ.FLAG) GO TO 310
C
C     ACCEPT ONLY ECHOES BETWEEN ZMIN AND ZMAX
C
      IF (SPBLOCK(1).LT.ZMIN) GO TO 30
      IF (SPBLOCK(1).GT.ZMAX) GO TO 30
C
C     REJECT ANGLES GREATER THAN 90 DEGREES
C
      IF (ABS(SPBLOCK(3)).GT.NINETY) GO TO 31
      IF (ABS(SPBLOCK(4)).GT.NINETY) GO TO 31
      GO TO 32
C
   31 NBAD=NBAD+1
      GO TO 30
C
  310 MY=SPBLOCK(2)
      MO=SPBLOCK(3)
      JO=SPBLOCK(4)
      LTIMH=SPBLOCK(5)
      LTIMM=SPBLOCK(6)
      MSEC=SPBLOCK(7)
C
C     CHANGE UT TO LOCAL MEAN SOLAR (UT-4H 28M)
C

      LTIMM=LTIMM-28
      IF(LTIMM.GE.0) GO TO 130
```

```
      LTIMM=LTIMM+60
      LTIMH=LTIMH-1
  130 LTIMH=LTIMH-4
      IF(LTIMH.GE.0) GO TO 195
      LTIMH=LTIMH+24
      JO=JO-1
  195 SPBLOCK(4)=JO
      SPBLOCK(5)=LTIMH
      SPBLOCK(6)=LTIMM
      SPBLOCK(7)=MSEC
C
C     THIS IS A USEABLE POINT!
C     WRITE SPBLOCK TO DISC
C
   32 WRITE (26) SPBLOCK
      IF(SPBLOCK(1).EQ.FLAG) GO TO 320
      NGOOD=NGOOD+1
      GO TO 30
  320 LOOP=LOOP+1
      IF(LOOP.NE.1) GO TO 30
      WRITE (*,*) SPBLOCK
      WRITE (17,*) SPBLOCK
      GO TO 30
C
C     CLOSE DISC FILE OF CURRENT NTAPE
C
   33 WRITE (*,100) NBAD,NGOOD
  100 FORMAT (/" NUMBER OF BAD RECORDS THIS TAPE =",I6/
     *" NUMBER OF USEABLE RECORDS THIS TAPE =",I6/)
      WRITE (17,100) NBAD,NGOOD
      WRITE (*,*) "             ***** EOF ***** EOF ***** EOF *****"
      WRITE (17,*) "             ***** EOF ***** EOF ***** EOF *****"
      WRITE (*,*) " "
      WRITE (17,*) " "
      NBAD=NOUGHT
      CLOSE (26)
      CLOSE (16,STATUS="DELETE")
   34 NTAPE=NTAPE+1
      IF (NTAPE.GT.LAST) GO TO 20
      CALL IBAD (NTAPE,SKIP)
      IF (SKIP.EQ.1) JERR=IQTAPE (7,CHAN,IFIX,SKIP,
     *BITE,SIZE,NBOUT,STATS)                                !SKIP FILE
      IF (SKIP.EQ.1) WRITE (*,*) " SKIPPING FILE SPP - GR",NTAPE
      IF (SKIP.EQ.1) WRITE (17,*) " SKIPPING FILE SPP - GR",NTAPE
      IF(JERR.EQ.8) GO TO 20                                 !END OF
TAPE MARKER
      IF (SKIP.EQ.1) GO TO 34
      GO TO 6
C
C     PROCESSING COMPLETE
C
   20 CLOSE (17)
      PAUSE " ALL DONE"
      STOP
      END
      SUBROUTINE RDYWANG (SKIP)
C                                  DECEMBER 13, 1991
C     SCATTERING POINT PARAMETERS - WangDAT TAPE SETUP
```

```
C
C
      CHARACTER*40 DFILE,INSTART,INEND
      integer*1 buf(1),stats(64)
      INTEGER*4 CHAN,SIZE,SKIP,SKIPIT,TRY,TAPEONE
C     ALL INTEGERS MUST BE LONGWORD (INTEGER*4)
      COMMON STATS,BUF
C
      WRITE (*,*) " READYING WangDAT TAPE DRIVE"
      WRITE (*,*) " "
      CHAN=3
      IFIX=0
      SKIPIT=0
      NUM=1
      SIZE=16384
      NBOUT=1
      NFILE=0
      TRY=0
C
      WRITE (*,*) " "
    1 WRITE (*,*) " TAPE DRIVE POSITIONING - PLEASE WAIT."
        jerr=iqtape(0,chan,ifix,skipit,buf,size,nbout,stats)       !UNIT
READY?
      TRY=TRY+1
      IF(TRY.GT.2) GO TO 2
      IF(JERR.NE.0) GO TO 1
        jerr=iqtape(5,chan,ifix,skipit,buf,size,nbout,stats)     !rewind
the tape
      IF(JERR.NE.0) GO TO 2
      jerr=iqtape(7,chan,ifix,SKIP,buf,size,nbout,stats)     !skip SKIP
files
      RETURN
    2 WRITE (*,104) JERR
  104 format (" ERROR ",I3,"  CHECK WANGDAT DRIVE")
      WRITE (25,104) JERR
      PAUSE " HIT RETURN TO EXIT"
      STOP
      END
      SUBROUTINE SWITCH7 (BLOCK,BYTE)
C                                                   MAY 15, 1991
C     SWITCHES TO AND FROM IBM/MAC FLOATING POINT NUMBERS.
C
      CHARACTER*1 A(4),I(4)
      REAL BLOCK(10),BYTE(10)
      EQUIVALENCE (R,A)
      DO 1 J=1,10
      R=BYTE(J)
      I(1)=A(4)
      I(2)=A(3)
      I(3)=A(2)
      I(4)=A(1)
      A(1)=I(1)
      A(2)=I(2)
      A(3)=I(3)
      A(4)=I(4)
      BLOCK(J)=R
    1 CONTINUE
      RETURN
```

```
      END
      SUBROUTINE IBAD (IFILE,SKIP)
C
C     SETS SKIP TO 1 IF IFILE IS BAD
C
C     THIS ROUTINE FOR AIDA'89 APRIL 5-11,
C     AND MAY 2 - 9 (NO BAD FILES?), 1989 ONLY.
C
      INTEGER*4 BADFILES(8),SKIP
      DATA BADFILES/168,218,223,233,245,247,249,250
      SKIP=0
      DO 1 I=1,8
      IF(IFILE.EQ.BADFILES(I)) GO TO 2
    1 CONTINUE
      RETURN
    2 SKIP=1
      RETURN
      END
```

# The tidal wind analysis program GROVES.f

The FORTRAN 77 program GROVES calculates the zonal, meridional and vertical components of the mean (prevailing) wind and periodic components (up to 4 harmonics, including the fundamental) from the scattering point parameter files SPP - GR XXX on disc. The usual configuration (see input file 7GRODAT below) selects the diurnal (24hr) and semidiurnal (12hr) tidal components, but any fundamental period (less than the data interval, of course!) may be specified.

Requires the following input files in the same folder

CADDSPEC, which specifies the designator for the output files. Specification is year, month, startday of interval to be analysed.

```
890503
XXXXXX
```

7DUNK, which contains the header for each page of the output file XXXXXXGROOUT.

```
 RESULTS FOR     APRIL  5 -     APRIL 11   1989      MAPSTAR
AIDA '89
000000000000000000000000000000000000000000000000000000000000000000000000
00000000
```

7DATES, which contains date and location information.

```
000000000000000000000000000000000000000000000000000
    APL  5    APL 11   1989
 ARECIBO (18N,67W)
0000XXX0XX000XXX0XX00XXXX000000X0000000000000000000
```

7GRODAT, which determines the processing parameters as follows

```
890405.   890411.        INCLUSIVE DATES
     0                   START HOUR
    5.   10.             ZENITH ANGLES ACCEPTED
     0                   RANGE CORRECTION
  2  2  2                NO. OF HARMONICS (INCLUDING FUNDAMENTAL)
  66.116.                HEIGHT RANGE
    24.                  FUNDAMENTAL PERIOD
  5  5  5  5  5  5       POLYNOMIAL FIT COEFFICIENTS NA
  5  5  5  5  5  5                                   NB
  3  3  3  3  3  3                                   NC
  0                      PROCESS AZIMUTH QUADRANTS 1-4  (0 FOR ALL
QUADRANTS)
```

The subroutines used are listed as include files in the main program. The purpose of each is detailed as comments in each program.

The output files , with prefixes as in CADDSPEC above, are

ECHORATE, a height/time table of scattering point rate

GLDPRNT, a report format table of wind components

ARCHIT, a binary file used internally

DIAGS, a file of runtime diagnostics

GROOUT, contains all information appropriate to processing, including a listing of all SPP - GR XXX files read, an echo rate map and tables of zonal, meridional and vertical wind components

TIDE, a binary file containing the tidal amplitudes and phases

ERROR, a binary with the tidal amplitude and phase errors and

ATIDE, an ASCII version of TIDE.

```
000000.000000000.X000000000000000000000000000000000000000000000000000000000
0000
C     PROGRAM GROSTAR
C                                                   SEPTEMBER 17, 1992
C     GROVES ANALYSIS  - MACINTOSH II MPW VERSION,
C                                 WITH INCLUDE FILES.
C
C     APPLIES STANDARD GROVES METEOR WIND ANALYSIS TO MAPSTAR RADAR
C     SCATTERING POINT PARAMETER DATA AS SPP - GR XXX FILES
C     WRITTEN ON DISC BY PROGRAM TAPEREAD.
C
C     A RUN CONSISTS OF
C     MAIN PROGRAM GROSTAR.apl
C     WHICH CONTAINS AS APPENDED FILES
C           SUBROUTINE MAPARAM
C           SUBROUTINE MAPSTAR
C           SUBROUTINE MATSIN
C           SUBROUTINE GMONTH
C           SUBROUTINE DAYMON
C           SUBROUTINE SDIANA
C           SUBROUTINE SVARY
C           SUBROUTINE SDESIG
C           SUBROUTINE ECHO
C           SUBROUTINE GROWZ
C           SUBROUTINE GWINGZ
C           SUBROUTINE GPRINT7
C           SUBROUTINE TRANS7
C            SUBROUTINE IBAD.f
C            SUBROUTINE DateTime
C
C     FILES XXXXARCHIT, XXXXECHORATE, XXXXERROR AND XXXXTIDE
C     (FOR INPUT TO ARCHIVE), XXXXATIDE (FOR INPUT TO ARCH.ETC)
C     XXXXZONAL AND XXXXMERIDIONAL (FOR INPUT TO "WINGZ"),
C     AND XXXXGLDPRNT (REPORT FORMAT WIND COMPONENT LISTING)
C     ARE CREATED BY  GROVES.
C     XXXX DENOTES THE FILE SDESIGNATOR FOR ANY GIVEN RUN, INPUT
C     AT RUNTIME FROM AUTOMATICALLY UPDATED HDISC FILE CADDSPEC.
C
C     READS RESULT SOURCE FROM FILE 7DUNK (INPUT FILE, HDISC) FORMAT
72A1,8A1
C     READS INPUT DATA FROM 7GRODAT (INPUT FILE, HDISC)
C     IN THE FOLLOWING ORDER
C     DATA INTERVAL TO BE PROCESSED,  FORMAT I6,3X,I6
C     STRTDA = START DAY,  6 DIGITS - YEAR MONTH DAY.  690307 IS MARCH
7,1969
C     ENDDAY = END DAY,     6 DIGITS - YEAR MONTH DAY.  700529 IS MAY
29,1970.
C     IF STRTDA-ENDDAY IS BLANK, ALL DATA IN FILE WILL BE PROCESSED.
C     START HOUR,                         FORMAT I6
C     IF START HOUR > 0, ONLY 24 HOURS OF DATA WILL BE PROCESSED.
C     ZENITH ANGLES ACCEPTED,             FORMAT 2F6.0
C     RANGE CORRECTION,                   FORMAT F6.0
C     EAST-WEST TIME VARIATION NP, NORTH-SOUTH TIME VARIATION NQ,
C     AND VERTICAL TIME VARIATION NR, FORMAT 3I3
C     PERIODICITY OF FUNDAMENTAL TIME VARIATION FORMAT F7.0
C     HEIGHT RANGE ZMIN, ZMAX. FORMAT I5,I4
C     EAST-WEST HEIGHT PROFILE.   FORMAT 24I3
```

```
C     NORTH-SOUTH HEIGHT PROFILE. FORMAT 24I3
C     VERTICAL HEIGHT PROFILE.    FORMAT 24I3
C     WINDS ARE CONSIDERED HORIZONAL IF VERTICAL HEIGHT PROFILE IS
C     SDESIGNATED NEGATIVE
C     QUADRANTS TO BE PROCESSED - ZERO FOR ALL QUADRANTS.  FORMAT I3
C     SUBROUTINE GPRINT7 READS DATA INTERVAL FROM 7DATES (INPUT FILE,
HDISC)
C     MONTH, DAY, - MONTH, DAY, YEAR, FORMAT  3X,A4,I3,2X,A4,I3,I5
C     REQUIRES SCATTERING POINT PARAMETER DATA ON WANGDAT TAPE (AS
C     FILE(S) "DUMMY" WHICH IS(ARE) SPECIFIED AT RUNTIME.
C     DATA MUST BE IN SPP OUTPUT TAPE FORMAT - 10 REALS PER RECORD
C     IN IBM PC REAL FORMAT, COMPLETE WITH TIMING FRAMES.
C
      GLOBAL DEFINE
      INCLUDE "Types.inc"
      INCLUDE "OSUtils.inc"
      END
C
      DIMENSION Q(200,200),P(200),AC(200),NA(10),NB(10),NC(10),
     1D(200),SIGMA(200),SINJ(10),COSJ(10),NTIME(24),A(200,200)
      CHARACTER*1 NGO,NPRINT,RESULT(72),SOURCE(8),FF
      CHARACTER*1 CADD(6)
      CHARACTER*40 ECHORATE,DUMMY,INSTART,INEND,TFILE,T4
      INTEGER*1 ICADD(6),ZERO
      INTEGER*2 STRTHR
      INTEGER*4 M,IDATE(3),ISEC,ISEC1,ISEC2,ISEC3
      COMMON/WINDS/ N,Q,NOP,ZMIN,MIN,ZMAX,MAX,NA,NB,NC,NAO,NBO,
     *NCO,SUM,AC,NTIME,NP,NQ,NR,RESULT,SOURCE,PERIOD,
     *FF,CADD,Z,A
      COMMON/ECHOES/ P,VEL,SINJ,COSJ,DCL,DCM,DCN,SUM1,SUM2,SUM3,
     *D,SIGMA,AZENMIN,AZENMAX,IEND
      COMMON/EXTRAS/ IUNIT,ICADD,NGO,STRTDA,ENDDAY,LEAPYR,JMO,
     *MNO(20,24),NFILE,LENGTH,NEG,NPMAX,ZERO,NPRINT,DAY,JOBAD
      COMMON/GENE/ M,UR,MY,MO,JO,LTIMH,LTIMM,MSEC,EL3,EM3
      COMMON/FSPEC/IFILE,DUMMY,INSTART,INEND,NSTART,NPOINTS
      COMMON/HRSTRT/ STRTHR,RNGCOR,ISEC1,ISEC2,ISEC3
      COMMON/HEIGHTS/NOZ,NQUAD
      COMMON/TEMPO/IDATE,LHOUR,LMIN,LSEC

C
      CALL DateTime
      ISEC1=3600*LHOUR+60*LMIN+LSEC
C
C     SET UP PROCESSING PARAMETERS
C
      NBOMB=1
      CALL MAPARAM
C
C     READ DATA, ECHO BY ECHO.
C
      M=0
      UR=0.0
      JOBAD=0
 3    CALL MAPSTAR
      IF (UR.LT.0.9) GO TO 100
C
C     NEXT COMES PROCESSING OF ECHO DATA TO PRODUCE COLUMNS D AND P,
C     AND MATRIX Q.
```

```
C
      DCL=EL3
      DCM=EM3
      DCN=SQRT(1.0-EL3**2-EM3**2)
C
C     CHANGE DIRECTION COSINES AND VELOCITY IF WIND IS TO BE CONSIDERED
C     HORIZONTAL
C
      IF(NEG.NE.-1) GO TO 333
      SINKI=SQRT(1.0-DCN**2)
      DCL=DCL/SINKI
      DCM=DCM/SINKI
      DCN=0.0
      VEL=VEL/SINKI
C
 333  CONTINUE
      CALL ECHO
      GO TO 3
C
C     PRELIMINARY OUTPUT.
C

  100 NBOMB=2
      CLOSE (IUNIT)
      IF (NQUAD.NE.0) GO TO 198
      WRITE (26,197)
  197 FORMAT (/"  ALL QUADRANTS PROCESSED"/)
      GO TO 150
  198 WRITE (26,199) NQUAD
  199 FORMAT(/"  QUADRANT",I2," PROCESSED",/)
  150 ENDDAY=DAY
      WRITE(26,200) STRTDA,ENDDAY
 200  FORMAT(/15H  DATA INTERVAL,3X,F7.0,4H  TO,F8.0//
     11X," VARIATION OF UPPER ATMOSPHERE WINDS WITH HEIGHT",/
     2/1X," GROVES ANALYSIS, WITH ERROR DETERMINATION"//)
      WRITE(26,201) CADD
  201 FORMAT(1X," OUTPUT FILES FROM THIS RUN HAVE PREFIX ",4A1)
      WRITE(26,202) M,NSTART,NPOINTS,AZENMIN,AZENMAX,N,INSTART,
     *INEND,STRTHR,NP,NQ,NR,MAX,MIN,NAO,NA,NBO,NB,NCO,NC,PERIOD
 202  FORMAT(1X//1X," NUMBER OF SCATTERING POINTS PROCESSED   =",
     1I8/"  (STARTING FROM ",I5," AND PROCESSING ",I5," POINT(S)
     2",/"  FOR EACH 1 KM HEIGHT INTERVAL OF EACH RADAR FRAME",
     3/"  WITH ZENITH ANGLES BETWEEN",F4.0,"AND",F4.0," DEGREES)"/
     4/1X," NUMBER OF INPUT PARAMETERS    =",I4,///1X," DATA READ
     5 FROM TAPE FILES STARTFILE   ",A40/24X," TO ENDFILE     ",A40,/
     *1X," STARTING AT HOUR ",I3/
     6/1X," TIME SERIES PARAMETERS P =",I4,"  Q =",I4,"  R =",I4,/
     7/1X," HEICHT RANGE, MAXIMUM  ",I5,1X,"  MINIMUM  ",I5,///1X,
     8" POWER SERIES PARAMETERS"///29X,"NA",11I3//29X,"NB",11I3/
     9/29X,"NC",11I3////1X," PERIOD",F7.1," HOURS")
C
C    SCATTERER RATE AS A FUNCTION OF TIME AND HEIGHT.
C
      NOP=NOP+1
      WRITE (26,999) FF, RESULT,SOURCE,NOP
  999 FORMAT(A1/72A1,8A1,26X,"PAGE",I3)
      WRITE(26,4000) NTIME
 4000 FORMAT(//" SCATTER RATE AS A FUNCTION OF TIME AND HEIGHT."//)
```

```
     11X,7H HEIGHT,24I5/1X)
      NH=(MAX-MIN)/3+1
      DO 4002 K=1,NH
      NZ=MAX-3*K+3
      KK=NH-K+1
      WRITE(26,4001) NZ,(MNO(KK,J),J=1,24)
 4001 FORMAT(1X,I5,2X,24I5/)
C
C      FILE ECHO RATE MAP IN "ECHORATE"
C
      ECHORATE="ECHORATE "
      CALL SDESIG (ECHORATE,CADD)
      OPEN (12,FILE=ECHORATE,FORM="UNFORMATTED")
      WRITE (12) (MNO(KK,J),J=1,24)
 4002 CONTINUE
      CLOSE (12)
C
      IF(M.GT.120) GO TO 400
      NBOMB=3
      GO TO 1102
C
C     INVERSION OF Q, AND FORMATION OF COEFFICIENT COLUMN AC.
C
 400  CONTINUE
      DO 101 J=1,N
      DO 101 K=1,N
      A(J,K)=Q(J,K)
 101  CONTINUE
      NBOMB=4
      WRITE (*,*) M," POINTS PROCESSED"
      WRITE (*,*) " "
      WRITE (*,*) " ATTEMPTING INVERSION OF MATRIX Q (TO A) "
      WRITE (*,*) " "
      CALL MATSIN(A,N,DETERM)
      IF(DETERM.GT.-12.0) GO TO 1103
 1102 WRITE(25,1104) FF, RESULT,N,N,DETERM,NBOMB
 1104 FORMAT(A1////1X,72A1////1X,52H **** ERROR IN INPUT DATA HAS
RESULTED
     1 IN MATRIX  Q(I3,1H,I3,34H) BEING UNSUITABLE FOR INVERSION.////
     2 10X,50H     $$$$$     $$$$$     $$$$$     $$$$$     $$$$$/////
     31X,13H DETERMINANT ,E12.4///1X,18H CONTINGENCY LEVEL,I5///
     41X,30H PROGRAMME CANNOT BE CONTINUED)
      PAUSE " NON-INVERTABLE MATRIX Q  -  CHECK DATES IN 7GRODAT!"
      GO TO 302
C
C     FORMULATE MODEL COEFFICIENTS ( AC )
C
 1103 CONTINUE
      DO 103 K=1,N
      DO 103 J=1,N
      AC(K)=AC(K)+P(J)*A(J,K)
 103  CONTINUE
      DO 104 J=1,N
      DO 104 K=1,N
      SUM1=SUM1+AC(J)*AC(K)*Q(J,K)
 104  CONTINUE
      DO 105 J=1,N
      SUM2=SUM2+AC(J)*P(J)
```

```
 105  CONTINUE
      M10=M 10
      SUM=(SUM1-(2.0*SUM2)+SUM3) FLOAT(M10 N)
      DO 106 J=1,N
      CHECK=A(J,J)*SUM
      ACHECK=ABS(CHECK)+0.01
      SIGN=CHECK/ACHECK
 106  SIGMA(J)=SIGN*SQRT(ACHECK)
C
C     OUTPUT
C
      NOP=NOP+1
      WRITE (26,999) FF, RESULT,SOURCE,NOP
      WRITE(26,35) DETERM,N
 35   FORMAT(/36H LOG (BASE 10) OF MATRIX DETERMINANT,E13.6
     *//" COLUMN MATRIX AC(",I3,")"/1X)
      LINE=0
      MA=N/2+1
      DO 205 I=1,MA
 208  WRITE(26,209) AC(I),SIGMA(I),AC(I+MA),SIGMA(I+MA)
 209  FORMAT(13X,2(13X,F7.2,3X,F7.1))
      LINE=LINE+1
      IF (LINE.LT.42) GO TO 205
      LINE=0
      NOP=NOP+1
      WRITE (26,999) FF, RESULT,SOURCE,NOP
      WRITE(26,36) N
 36   FORMAT(1X//" COLUMN MATRIX AC(",I3,") (CONTD) "/1X)
 205  CONTINUE
      WRITE(25,*) " AC RECORDED"
      IF (PERIOD.NE.24.0) GO TO 300
      WRITE (*,*) "    CALLING SDIANA"
      CALL SDIANA
      WRITE (*,*) "    DIURNAL VARIATION COMPLETED"
      WRITE (25,*) "    DIURNAL VARIATION COMPLETED"
 300  WRITE (*,*) "    CALLING SVARY"
      CALL SVARY
      WRITE (*,*) "    TIDES FILED "
      WRITE (25,*) "    TIDES FILED "
      IF (PERIOD.NE.24.0.OR.NPMAX.GT.2) GO TO 301
C
C     PREPARE FILE "GLDPRNT"
C
      CALL GPRINT7
C
C     PREPARE INPUT FILES FOR "WINGZ"
C
      CALL GROWZ (CADD)
      WRITE (25,*) " WINGZ FILES COMPLETED"
C
C     PREPARE SPYGLASS TRANSFORM PLOT FILES
C
      TFILE="UUU"
      T4="T4U "
      CALL TRANS7 (TFILE,T4)
      TFILE="VVV"
      T4="T4V "
      CALL TRANS7 (TFILE,T4)
```

```
      T4="T4W "
      TFILE="WWW"
      CALL TRANST (TFILE,T4)
      WRITE (25,*) " SPYGLASS FILES COMPLETED"
      WRITE (*,*) " SPYGLASS FILES COMPLETED"
C
  301 CALL DateTime
      ISEC2=3600*LHOUR+60*LMIN+LSEC
      ISEC=ISEC2 ISEC3-ISEC1
      JHOUR=ISEC/3600
      JMIN=ISEC/60-JHOUR*60
      JSEC=ISEC-JHOUR*3600-JMIN*60
      WRITE (*,107) JHOUR,JMIN,JSEC,M
  107 FORMAT (///" PROGRAM TOOK ",I3," HOURS ",I2," MINS ",I2," SECS
     * TO PROCESS ",I6," ECHOES" A1)
      WRITE (25,107) JHOUR,JMIN,JSEC,M
      WRITE (26,107) JHOUR,JMIN,JSEC,M,FF
      WRITE (25,108)
  108 FORMAT (" SUCCESSFUL COMPLETION OF GROVES ANALYSIS!")
      CLOSE (25)
      CLOSE (26)
      WRITE (*,*) " "
      WRITE (*,108)
      WRITE (*,*) " "
      WRITE (*,*) " "
      WRITE (*,*) " ISN'T THIS FUN?"
      WRITE (*,*) " "
      WRITE (*,*) " "
      WRITE (*,*) "                          - HIT RETURN TO EXIT"
      PAUSE
      STOP
  302 WRITE (*,*) " "
      PAUSE " ERROR EXIT"
      WRITE (25,109)
  109 FORMAT (" ERROR EXIT. STATEMENT 302")
      WRITE (26,109)
      CLOSE (25)
      CLOSE (26)
      STOP
      END



      SUBROUTINE DateTime
      IMPLICIT NONE
      integer*4 month,day,year,hour,minute,second
      COMMON/TEMPO/year,month,day,hour,minute,second
C define a datetime record, structure and comments taken
C from Inside MacIntosh, Vol 2, page 378

      RECORD /DateTimeRec/ DateTime

      CALL GetTime(DateTime)
        month=DateTime.month
        day=DateTime.day
        year=DateTime.year
        hour=DateTime.hour
```

```
  minute=DateTime.minute
second=DateTime.second
  RETURN
  END
```

```
      SUBROUTINE DAYMON(MYRDAY,L,MO,JO)
C
C                                    MARCH [illegible]
C     CALCULATES DAY OF YEAR RELATIVE TO ZERO HOUR, JANUARY 1
C
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12) MO
    1 MYRDAY=JO
      RETURN
    2 MYRDAY=31+JO
      RETURN
    3 MYRDAY=59+JO+L
      RETURN
    4 MYRDAY=90+JO+L
      RETURN
    5 MYRDAY=120+JO+L
      RETURN
    6 MYRDAY=151+JO+L
      RETURN
    7 MYRDAY=181+JO+L
      RETURN
    8 MYRDAY=212+JO+L
      RETURN
    9 MYRDAY=243+JO+L
      RETURN
   10 MYRDAY=273+JO+L
      RETURN
   11 MYRDAY=304+JO+L
      RETURN
   12 MYRDAY=334+JO+L
      RETURN
      END
```

```
      SUBROUTINE ECHO
C                                               MAY 14, 1991
C     SETS UP MATRIX EQUATION BY PROCESSING  UFF DATA FILE
C
      SAVE
      DIMENSION NA(10),NB(10),NC(10),AC(200),SUMSA(10),SUMSB(10),
     *SUMSC(10),D(200),P(200),Q(200,200),NTIME(24),SINJ(10),COSJ(10)
     *,SIGMA(200)
      CHARACTER*1 CADD(6),RESULT(72),SOURCE(8),FF
      COMMON/WINDS/ N,Q,NOP,ZMIN,MIN,ZMAX,MAX,NA,NB,NC,NAO,NBO,
     *NCO,SUM,AC,NTIME,NP,NQ,NR,RESULT,SOURCE,PERIOD,
     *FF,CADD,Z
      COMMON/ECHOES/ P,VEL,SINJ,COSJ,DCL,DCM,DCN,SUM1,SUM2,SUM3,
     *D,SIGMA
C
C     CALCULATE NORMALIZED HEIGHT OF SCATTERING POINT
C
      S=(2.0*Z-ZMAX-ZMIN)/(ZMAX-ZMIN)+1.0E-06
C
      SUMP=0.0
      SUMQ=0.0
      SUMR=0.0
      NCOUNT=0
 1000 NAOT=2*NAO
      SUMSAO=1
      IF(NAOT) 110,110,84
 84   DO 8 K=2,NAOT,2
      SUMSAO=SUMSAO+S**K
 8    CONTINUE
      IF(NP) 110,110,85
 85   DO 10 J=1,NP
      NA2=2*NA(J)
      SUMSA(J)=1
      DO 9 K=2,NA2,2
 9    SUMSA(J)=SUMSA(J)+S**K
      SUMP=SUMP+SUMSA(J)
 10   CONTINUE
 110  SUMP=SUMP+SUMSAO
      NBOT=2*NBO
      SUMSBO=1
      IF(NBOT) 130,130,114
 114  DO 11 K=2,NBOT,2
      SUMSBO=SUMSBO+S**K
 11   CONTINUE
      IF(NQ) 130,130,115
 115  DO 13 J=1,NQ
      NB2=2*NB(J)
      SUMSB(J)=1
      DO 12 K=2,NB2,2
 12   SUMSB(J)=SUMSB(J)+S**K
      SUMQ=SUMQ+SUMSB(J)
 13   CONTINUE
 130  SUMQ=SUMQ+SUMSBO
      NCOT=2*NCO
      SUMSCO=1
      IF(NCOT) 160,160,135
 135  DO 14 K=2,NCOT,2
      SUMSCO=SUMSCO+S**K
```

```
14    CONTINUE
      IF(NR) 160,160,145
145   DO 16 J=1,NR
      NC2=2*NC(J)
      SUMSC(J)=1.0
      DO 15 K=2,NC2,2
15    SUMSC(J)=SUMSC(J)+S**K
      SUMR=SUMR+SUMSC(J)
16    CONTINUE
160   SUMR=SUMR+SUMSCO
      WF=1.0/((DCL**2)*SUMP+(DCM**2)*SUMQ+(DCN**2)*SUMR)
      SUM3=SUM3+WF*VEL**2
      NAOE=NAO+1
      DO 20 K=1,NAOE
      NCOUNT=NCOUNT+1
20    D(NCOUNT)=DCL*(S**(K-1))
      IF(NP) 322,322,320
320   DO 21 J=1,NP
      NAE=NA(J)+1
      DO 21 K=1,NAE
      NCOUNT=NCOUNT+1
21    D(NCOUNT)=DCL*(S**(K-1))*SINJ(J)
      DO 22 J=1,NP
      NAE=NA(J)+1
      DO 22 K=1,NAE
      NCOUNT=NCOUNT+1
22    D(NCOUNT)=DCL*(S**(K-1))*COSJ(J)
322   NBOE=NBO+1
      DO 23 K=1,NBOE
      NCOUNT=NCOUNT+1
23    D(NCOUNT)=DCM*(S**(K-1))
      IF(NQ) 325,325,323
323   DO 24 J=1,NQ
      NBE=NB(J)+1
      DO 24 K=1,NBE
      NCOUNT=NCOUNT+1
24    D(NCOUNT)=DCM*(S**(K-1))*SINJ(J)
      DO 25 J=1,NQ
      NBE=NB(J)+1
      DO 25 K=1,NBE
      NCOUNT=NCOUNT+1
25    D(NCOUNT)=DCM*(S**(K-1))*COSJ(J)
325   NCOE=NCO+1
      DO 26 K=1,NCOE
      NCOUNT=NCOUNT+1
26    D(NCOUNT)=DCN*(S**(K-1))
      IF(NR) 328,328,326
326   DO 27 J=1,NR
      NCE=NC(J)+1
      DO 27 K=1,NCE
      NCOUNT=NCOUNT+1
27    D(NCOUNT)=DCN*(S**(K-1))*SINJ(J)
      DO 28 J=1,NR
      NCE=NC(J)+1
      DO 28 K=1,NCE
      NCOUNT=NCOUNT+1
28    D(NCOUNT)=DCN*(S**(K-1))*COSJ(J)
328   DO 29 J=1,N
```

```
      P(J)=P(J)+WF*VEL*D(J)
 29   CONTINUE
      DO 30 J=1,N
      DO 30 K=1,N
      Q(J,K)=Q(J,K)+WF*D(J)*D(K)
 30   CONTINUE
      RETURN
      END




      SUBROUTINE GMONTH (RMONTH,I)
C                                 MARCH 30, 1990
C     DETERMINES HOLERITH MONTH
C
      CHARACTER*4 MO(12),RMONTH
      INTEGER*4 I
      DATA MO/" JAN"," FEB"," MAR"," APL"," MAY"," JUN"," JLY",
     *" AUG"," SEP"," OCT"," NOV"," DEC"/
      IF(I.LT.1.OR.I.GT.12) GO TO 1
      RMONTH=MO(I)
      RETURN
    1 WRITE(*,100) I
  100 FORMAT("ILLEGAL MONTH ",I5)
      PAUSE
      STOP
      END
```

```
      SUBROUTINE GPRINT7
C                                                         MAY 14, 1991
C     FOR USE WHEN ONLY MEAN, DIURNAL AND SEMIDIURNAL GROVES
C     FITS ARE USED TO GENERATE FILE "XXXXTIDE".
C
C     PREPARES FILE OF GROVES OUTPUT FOR PRINTING
C     IN REPORT STYLE FORMAT, WITH OUTPUT AT 4KM INTERVALS
C     FOR ZONAL, MERIDIONAL AND VERTICAL COMPONENTS.
C
C
C     ACCESSES FILES "XXXXTIDE" AND "XXXXERROR" (OUTPUT FROM JUARY)
C     AND "7DATES", IF AVAILABLE
C
      DIMENSION IH(47),AM(765),ER(765),AO(153),AU24(153),PH24(153),
     *AU12(153),PH12(153),EO(153),ER24(153),ERPH24(153),ER12(153),
     *ERPH12(153)
      CHARACTER*1 FF,WHERE(32),RESULT(72),SOURCE(8)
      CHARACTER*4 CADD,FILE,FMON1,FMON2
      CHARACTER*40 GLDPRNT
      COMMON/WINDS/ N,Q(200,200),NOP,ZMIN,MIN,ZMAX,MAX,
     *NA(10),NB(10),NC(10),NAO,NBO,NCO,SUM,AC(200),
     *NTIME(24),NP,NQ,NR,RESULT,SOURCE,PERIOD,
     *FF,CADD
C
      MHITE=MAX-MIN+1
      NHITE=MHITE-4
      IHITE=15*MHITE
      OPEN (5,FILE="7DATES",FORM="FORMATTED")
      REWIND (15)
      REWIND (16)
      GLDPRNT="GLDPRNT "
      CALL SDESIG (GLDPRNT,CADD)
      OPEN (17,FILE=GLDPRNT,FORM="FORMATTED")
C
C     CHECK DATES
C
      WRITE (*,*) " LOOKING FOR FILE DATES."
C
      GO TO 11
C
C     ENTER DATA INTERVAL IF FILE 1DATES MISSING
C
    9 WRITE(*,98)
   98 FORMAT("ENTER DATA INTERVAL; MONTH,DAY,MONTH,DAY,YEAR,
     * WHERE (3X,A4,I3,2X,A4,I3,1X,I5/32A1)")
      READ(*,99,ERR=9) FMON1,JO1,FMON2,JO2,MYEAR,WHERE
   99 FORMAT(/3X,A4,I3,2X,A4,I3,1X,I5/32A1)
      GO TO 12
   11 READ(5,99,ERR=9) FMON1,JO1,FMON2,JO2,MYEAR,WHERE
      CLOSE (5)
C
C     INITIALIZE
C
   12 DO 10 I=1,NHITE
      IH(I)=MAX+1-I
   10 CONTINUE
C
C     READ AMPLITUDES, PHASES AND ERRORS
```

```
C
      FILE="ERROR"
      READ (16,END=14,ERR=13) (ER(I),I=1,IHITE)
      FILE="TIDE "
      READ (15,END=14,ERR=13) (AM(I),I=1,IHITE)
C
C     SET UP APPROPRIATE ARRAYS
C
      DO 8 I=1,IHITE,5
      J=(I/5)+1
      AO(J)=AM(I)
      AU24(J)=AM(I+1)
      PH24(J)=AM(I+2)
      AU12(J)=AM(I+3)
      PH12(J)=AM(I+4)
      EO(J)=ER(I)
      ER24(J)=ER(I+1)
      ERPH24(J)=ER(I+2)
      ER12(J)=ER(I+3)
      ERPH12(J)=ER(I+4)
    8 CONTINUE
C
C     OUTPUT EAST-WEST WIND COMPONENTS
C
      WRITE(17,100) CADD,WHERE
  100 FORMAT(1X,A4/12X,"EAST-WEST WIND COMPONENTS, ",2X,32A1/)
      WRITE(17,101) FMON1,JO1,FMON2,JO2,MYEAR
  101 FORMAT(12X,A4,I3," - ",A4,I3,I5)
      WRITE(17,102)
  102 FORMAT(38X,"24 HOUR                12 HOUR",/
     *12X,"HEIGHT  MEAN ERROR  AMP ERROR  PHI ERROR
     *  AMP ERROR  PHI ERROR"/)
      DO 1 I=7,NHITE,4
      WRITE(17,103) IH(I),AO(I),EO(I),AU24(I),ER24(I),PH24(I),ERPH24(I),
     *AU12(I),ER12(I),PH12(I),ERPH12(I)
  103 FORMAT(12X,I4,F8.0,F5.0,4(F6.0,F5.0)/)
    1 CONTINUE
C
C     OUTPUT NORTH-SOUTH WIND COMPONENTS
C
      KHITE=(NHITE-6)/4+1
      IF (KHITE.GT.6) WRITE (17,1040) FF
 1040 FORMAT (A1)
      WRITE(17,104) CADD,WHERE
  104 FORMAT(1X,A4/12X,"NORTH-SOUTH WIND COMPONENTS ",32A1/)
      WRITE(17,101) FMON1,JO1,FMON2,JO2,MYEAR
      WRITE(17,102)
      DO 2 I=7,NHITE,4
      J=I+MHITE
      WRITE(17,103) IH(I),AO(J),EO(J),AU24(J),ER24(J),
     *PH24(J),ERPH24(J),AU12(J),ER12(J),PH12(J),ERPH12(J)
    2 CONTINUE
C
C     OUTPUT VERTICAL WIND COMPONENTS
C
      IF (KHITE.GT.6) WRITE (17,1040) FF
      WRITE(17,105) CADD,WHERE
  105 FORMAT(1X,A4/12X,"VERTICAL WIND COMPONENTS, ",3X,32A1/
```

```
     *12X," NOTE - VELOCITIES ARE IN CM S")
      WRITE(17,101) FMON1,JO1,FMON2,JO2,MYEAR
      WRITE(17,102)
      DO 3 I=7,NHITE,4
      J=I+2*MHITE
      WRITE(17,103) IH(I),AO(J),EO(J),AU24(J),ER24(J),
     *PH24(J),ERPH24(J),AU12(J),ER12(J),PH12(J),ERPH12(J)
    3 CONTINUE
      WRITE(*,106)
      WRITE(25,106)
  106 FORMAT(" GLDPRNT FILED")
      CLOSE (15)
      CLOSE (16)
      WRITE (17,1043) FF
      CLOSE (17)
      RETURN
   13 WRITE(*,107) FILE
  107 FORMAT(" NO DATA IN FILE ",A5)
      STOP " ERROR TERMINATION"
   14 WRITE(*,108) FILE
  108 FORMAT(" ERROR IN DATA FILE ",A5)
      PAUSE " ERROR TERMINATION"
      STOP
      END
```

```
      SUBROUTINE GROWZ (CADD)
C                                                   SEPTEMBER 17, 1991
C      READS XXXXARCHIT  ARCHIVE) FILE
C
C     WRITES FILES XXXXZONAL, XXXXMERIDIONAL AND XXXXVERTICAL WINDS
C     FOR PLOTTING BY "WINGZ"
C
C     REQUIRES SUBROUTINES
C                                            GWINGZ
C                                            SDESIG
C     AND INPUT FILE XXXXARCHIT
C     WHERE XXXX IS THE FILE DESIGNATOR "CADD"
C
C     (NOTE THAT X IS A DUMMY BLOCK
C     INTO WHICH WINDS ABOVE 100KM AND BELOW 80KM ARE READ, SINCE
C     OUTPUT FILES ARE DEFINED ONLY BETWEEN 80 AND 100KM.)
C
      DIMENSION U(25,41),V(25,41),W(25,41),NZ(41),X(25,6)
      CHARACTER*1 TAB
      CHARACTER*4 CADD
      CHARACTER*40 ARCHIT,WZFILE,OUTFILE
      COMMON/HEIGHTS/NOZ
      WRITE (*,*) "       SUBROUTINE GROWZ- PREPARING WINGZ INPUT FILES"
      WRITE (*,*) "       FROM GROVES ZONAL, MERIDIONAL AND VERTICAL
     * OUTPUT (IN ARCHIT)"
      OPEN (15,FILE="UUU",FORM="FORMATTED")
      OPEN (16,FILE="VVV",FORM="FORMATTED")
      OPEN (17,FILE="WWW",FORM="FORMATTED")
      ARCHIT="ARCHIT "
      CALL SDESIG (ARCHIT,CADD)
      OPEN (18,FILE=ARCHIT,FORM="UNFORMATTED")
      ITAB=9
      TAB=CHAR(ITAB)
      N=0
   10 DO 11 I=1,6
      READ (18) NZ(I),(X(J,I),J=1,25)
   11 CONTINUE
      DO 1 I=1,NOZ
      IF (N.EQ.0) READ (18) NZ(I),(U(J,I),J=1,25)
      IF (N.EQ.1) READ (18) NZ(I),(V(J,I),J=1,25)
      IF (N.EQ.2) READ (18) NZ(I),(W(J,I),J=1,25)
    1 CONTINUE
      DO 2 I=1,4
      READ (18) KZ,(X(J,I),J=1,25)
    2 CONTINUE
      N=N+1
      IF(N.LE.2) GO TO 10
      CLOSE (18)
      DO 4 J=1,NOZ,2
      WRITE  (15,100) NZ(J),TAB,(U(I,J),TAB,I=1,24),U(25,J)
      WRITE  (16,100) NZ(J),TAB,(V(I,J),TAB,I=1,24),V(25,J)
      WRITE  (17,100) NZ(J),TAB,(W(I,J),TAB,I=1,24),W(25,J)
  100 FORMAT (I6,A1,24(F8.2,A1),F8.2)
    4 CONTINUE
      CLOSE (15)
      CLOSE (16)
      CLOSE (17)
      OUTFILE="ZONAL "
```

```
CALL SDESIG (OUTFILE,CADD)
WZFILE="UUU"
CALL GWINGZ (WZFILE,OUTFILE)
OUTFILE="MERIDIONAL "
CALL SDESIG (OUTFILE,CADD)
WZFILE="VVV"
CALL GWINGZ (WZFILE,OUTFILE)
OUTFILE="VERTICAL "
CALL SDESIG (OUTFILE,CADD)
WZFILE="WWW"
CALL GWINGZ (WZFILE,OUTFILE)
RETURN
END
```

```
      SUBROUTINE GWINGZ (INFILE,OUTFILE)
C                                                         MAY 11, 1991
C     REFORMATS INPUT FILE INFILE FOR INPUT TO "WINGZ"
C     AS FILE OUTFILE.
C
C     "WINGZ" REQUIRES EACH COLUMN OF DATA TO BE SEPARATED
C     BY A TAB.
C     A CARRIAGE RETURN MUST APPEAR AT THE END OF EACH DATA ROW.
C     A SPREADSHEET DATA FIELD (CELL) WILL BE CONSIDERED TEXT
C     IF IT CONTAINS AT LEAST ONE BLANK CHARACTER.
C     THE FIRST ROW OF A "WINGZ" SPREADSHEET CONTAINS X AXIS
C     SCALE DESIGNATORS. THE FIRST COLUMN, THE Y AXIS DESIGNATORS
C     (I.E. NUMBERS, NOT LABELS)
C     AN AXIS ROW (OR COLUMN) DESIGNATOR CELL MUST APPEAR BLANK
C     IF IT CONTAINS NOTHING BUT A BLANK CHARACTER.
C     INTEGER ROW (OR COLUMN) DESIGNATOR FIELDS MUST END WITH
C     A BLANK, OTHERWISE THEY MAY APPEAR AS FLOATING POINT NUMBERS.
C
C
      INTEGER*1 IHR,JHR
      INTEGER*2 IF,ITAB,LCHAR,NTOP,ISTOP,STRTHR
      INTEGER*4 NCHAR,LINES
      CHARACTER*1 BLANK,HR(14),HOUR(104),REFMT(250),TAB
      CHARACTER*40 OUTFILE,INFILE
      COMMON/HRSTRT/ STRTHR
      COMMON/HEIGHTS/NOZ
      OPEN(15,FILE=INFILE,FORM="UNFORMATTED")
      OPEN(16,FILE=OUTFILE,FORM="FORMATTED")
C     LINES=NO. OF LINES IN INPUT FILE.
C     NCHAR=NO. OF CHARACTERS PER LINE.
      LINES=NOZ/2+1
      NCHAR=232
      BLANK=" "
      I48=48
      ITAB=9
      TAB=CHAR(ITAB)
      IHR=STRTHR/10
      JHR=STRTHR-10*IHR
      HR(1)=CHAR(IHR+I48)
      IF(IHR.EQ.0) HR(1)=BLANK
      HR(2)=CHAR(JHR+I48)
      J=0
      DO 1 I=1,11,2
      J=J+1
      JSTRT=STRTHR+4*J
      IF(JSTRT.GE.24) JSTRT=JSTRT-24
      IHR=JSTRT/10
      JHR=JSTRT-10*IHR
      HR(I+2)=CHAR(IHR+I48)
      IF(IHR.EQ.0) HR(I+2)=BLANK
      HR(I+3)=CHAR(JHR+I48)
    1 CONTINUE
      DO 10 I=1,104,4
      HOUR(I)=BLANK
      HOUR(I+1)=BLANK
      HOUR(I+2)=BLANK
      HOUR(I+3)=TAB
   10 CONTINUE
```

```
      HOUR(5)=HR(1)
      HOUR(6)=HR(2)
      HOUR(21)=HR(3)
      HOUR(22)=HR(4)
      HOUR(37)=HR(5)
      HOUR(38)=HR(6)
      HOUR(53)=HR(7)
      HOUR(54)=HR(8)
      HOUR(69)=HR(9)
      HOUR(70)=HR(10)
      HOUR(85)=HR(11)
      HOUR(86)=HR(12)
      HOUR(101)=HR(13)
      HOUR(102)=HR(14)
      WRITE (*,*) "  PREPARING ",OUTFILE
      WRITE (16,100) (HOUR(IH),IH=1,103)
  100 FORMAT(103A1)
      DO 5 I=1,LINES
      READ (15) (REFMT(IC),IC=1,NCHAR)
      NTOP=NCHAR
      IF=7
    2 IF=IF+1
      IF(IF.EQ.ISTOP) GO TO 4
   20 IF(REFMT(IF).NE.BLANK) GO TO 2
      KTOP=NTOP-IF
      DO 3 KK=1,KTOP
      REFMT(IF-1+KK)=REFMT(IF+KK)
    3 CONTINUE
      NTOP=NTOP-1
      ISTOP=NTOP-1
      GO TO 20
    4 LCHAR=NTOP-1
      WRITE (16,101) (REFMT(IC),IC=1,LCHAR)
  101 FORMAT(250A1)
    5 CONTINUE
      CLOSE (15)
      CLOSE (16)
      RETURN
      END
```

```
      SUBROUTINE IBAD (IFILE,SWITCH)
C
C     SETS SKIP TO 1 IF IFILE IS BAD
C
C     THIS ROUTINE FOR AIDA'89 APRIL 6-11, 1989 ONLY.
C
      INTEGER*4 BADFILES(8),SKIP
      DATA BADFILES/168,218,223,233,245,247,249,251/
      SKIP=0
      DO 1 I=1,8
      IF(IFILE.EQ.BADFILES(I)) GO TO 2
    1 CONTINUE
      RETURN
    2 SKIP=1
      RETURN
      END
```

```
      SUBROUTINE MAPARAM
C                                          SEPTEMBER [illegible], 19[illegible]
C     SETS UP PROCESSING PARAMETERS PRIOR TO READING [illegible] MMM
      DATA FROM DISK
C
      DIMENSION Q([illegible]),[illegible],P([illegible]),D([illegible]),AZ([illegible]),
     *NA(10),NB(10),NC(10),SIGMA([illegible]),
     *[illegible](10),[illegible](10),NTIME(24)
      CHARACTER*1 MGO,NPRINT,RESULT([illegible]),SOURCE([illegible]),FF
      CHARACTER*1 CADD(6),DUMSTA(40),DUMEND(40)
      CHARACTER*4 KDATE
      CHARACTER*6 WHICH
      CHARACTER*40 GROOUT,DUMMY,INSTART,INEND,DIAGS
      INTEGER*1 ADD,ICADD(6),ZERO
      INTEGER*2 STRTHR
      INTEGER*4 IDATE(3)
      COMMON/HRSTRT/ STRTHR,RNGCOR
      COMMON/WINDS/ N,Q,NOP,ZMIN,MIN,ZMAX,MAX,NA,NB,NC,NAC,NBC,
     *NCO,SUM,AC,NTIME,NP,NQ,NR,RESULT,SOURCE,PERIOD,
     *FF,CADD,Z
      COMMON/ECHOES/ P,VEL,SINZ,COSZ,DCL,DCM,DCN,SUM1,SUM2,SUM3,
     *D,SIGMA,AZENMIN,AZENMAX,IEND
      COMMON/EXTRAS/ IUNIT,ICADD,MGO,STRTDA,ENDDAY,LEAPYR,JMO,
     *MNO(20,24),NFILE,LENGTH,NEG,NPMAX,ZERO,NPRINT
      COMMON/FSPEC/IFILE,DUMMY,INSTART,INEND,NSTART,NPOINTS
      COMMON/HEIGHTS/NOZ,NQUAD
      COMMON/TEMPO/IDATE,LHOUR,LMIN,LSEC
      EQUIVALENCE (DUMSTA,INSTART)
      EQUIVALENCE (DUMEND,INEND)
C
C     FILE ORGANIZATION
C
      WRITE (*,*) "              GROVES ANALYSIS"
      WRITE (*,*) " "
      WRITE (*,*) " READING OUTPUT FILE SDESIGNATOR"
      OPEN (25,FILE="CADDSPEC",FORM="FORMATTED")
      READ (25,98) ICADD
   98 FORMAT(6I1)
      CLOSE (25)
      ADD=48
      CADD(1)=CHAR (ADD+ICADD(1))
      CADD(2)=CHAR (ADD+ICADD(2))
      CADD(3)=CHAR (ADD+ICADD(3))
      CADD(4)=CHAR (ADD+ICADD(4))
      CADD(5)=CHAR (ADD+ICADD(5))
      CADD(6)=CHAR (ADD+ICADD(6))
      WRITE (*,99) ICADD,CADD
   99 FORMAT (4I1,3X,4A1)
      DIAGS="DIAGS "
      CALL SDESIG (DIAGS,CADD)
      OPEN (25,FILE=DIAGS,FORM="FORMATTED")
      OPEN (5,FILE="7GRODAT",FORM="FORMATTED")
      IUNIT=19
      ZERO=0
      FF=CHAR(12)
C
C     SPECIFY CENTURY
C
```

```
      MYR19=1900
      LEAPYR=0
C
C     SELECT DATA INTERVAL, START HOUR ETC. TO BE PROCESSED
C
      READ (5,32,END=299) STRTDA,ENDDAY
   32 FORMAT(F7.0,3X,F7.0)
       READ (5,33,END=299)STRTHR
   33 FORMAT(I6)
      READ (5,34,END=299) AZENMIN,AZENMAX
   34 FORMAT(2F6.0)
      READ (5,34,END=299) RNGCOR
      IF (ABS(RNGCOR).GT.10) PAUSE " BAD RANGE CORRECTION;
     *  TYPE COMMAND PERIOD TO EXIT "
      WRITE (*,35) STRTDA,ENDDAY,STRTHR,AZENMIN,AZENMAX,RNGCOR
   35 FORMAT(F7.0,3X,F7.0,3X,I6,3F6.1, " IS TO BE PROCESSED. CORRECT?")
      IMYMO=STRTDA/100
      IMY=IMYMO/100
      LEAP=(MYR19+IMY)/4
      LEAP=LEAP*4-(MYR19+IMY)
      IF(LEAP.EQ.0) LEAPYR=1
      IMO=IMYMO-IMY*100
      GROOUT="GROOUT "
      CALL SDESIG (GROOUT,CADD)
      OPEN (26,FILE=GROOUT,FORM="FORMATTED")
C
C     INITIALIZATION
C
      NBOMB=0
      NEG=0
      NRDERR=0
      DO 500 IT=1,24
      IHR=IT+STRTHR-1
      IF(IHR.GE.24) IHR=IHR-24
      NTIME(IT)=IHR
  500 CONTINUE
      DO 38 I=1,200
      P(I)=0.0
      D(I)=0.0
      SIGMA(I)=0.0
      AC(I)=0.0
      DO 38 L=1,200
      Q(I,L)=0.0
 38   CONTINUE
      NUMNA=0
      NUMNB=0
      NUMNC=0
      SUM1=0.0
      SUM2=0.0
      SUM3=0.0
      M=0
      NOP=0
      DO 39 I=1,10
      NA(I)=0
      NB(I)=0
      NC(I)=0
 39   CONTINUE
      DO 41 I=1,20
```

```
      DO 41 J=1,24
      MNO(I,J)=0
 41   CONTINUE
C
C     INTERROGATE MACHINE FOR DATE
C
      CALL DateTime
      IDATE(1)=IDATE(1)-1900
      CALL GMONTH (KDATE,IDATE(2))
      OPEN (16,FILE="7DUNK",FORM="FORMATTED")
C
C     READ AND PRINT FIRST PAGE HEADER
C
      READ(16,990) RESULT,SOURCE
 990  FORMAT(72A1,8A1)
      CLOSE (16)
      NOP=1
      WRITE(26,998) RESULT,SOURCE,KDATE,IDATE(3),IDATE(1),NOP
 998  FORMAT(////1X,72A1,8A1,4X,6HRUN ON,A4,I3," 19",I2,3X,"PAGE ",
     *I3////)
      WRITE(25,998) RESULT,SOURCE,KDATE,IDATE(3),IDATE(1),NOP
C
C     THIS SECTION OF THE PROGRAM READS PROCESSING PARAMETERS.
C
      NBOMB=NBOMB+1
      IF(STRTDA.GT.ENDDAY) GO TO 299
      IF(ENDDAY.EQ.0) GO TO 6
      JUMP=0
      JMYMO=ENDDAY/100
      IJO=STRTDA-IMYMO*100
      JJO=ENDDAY-JMYMO*100
      JMY=JMYMO/100
      JMO=JMYMO-JMY*100
      GO TO 7
C
 299  NBOMB=NBOMB+1
      WRITE(25,210) FF, NBOMB
 210  FORMAT(A1////1X,"EXIT, CONTIGENCY LEVEL",I2////1X,"NO DATA IN FI
     1LE 5")
      PAUSE "NO DATA IN FILE"
      STOP
C
 6    JUMP=1
      ENDDAY=1.0E+06
 7    CONTINUE
      READ(5,1) NP,NQ,NR
 1    FORMAT(24I3)
      NPMAX=NP
C
      IF(NQ.LE.NPMAX) GO TO 60
      NPMAX=NQ
 60   IF(NR.LE.NPMAX) GO TO 61
      NPMAX=NR
 61   CONTINUE
      READ(5,2) ZMIN,ZMAX
 2    FORMAT(1X,2F4.0)
      WRITE (*,*) ZMIN,ZMAX
      MIN=ZMIN
```

```
      MA?
      GO TO 7
C
 ???  NBOMB=NBOMB+1
      WRITE(25,210) FF, NBOMB
 210  FORMAT(A1//  1X,"EXIT, CONTIGENCY LEVEL",I?    1X,"NO DATA IN FI
     1LE 5")
      PAUSE "NO DATA IN FILE"
      STOP
C
 ?    JUMP=1
      ENDDAY=1.0E+06
 7    CONTINUE
      READ(5,1) NP,NQ,NR
 1    FORMAT(24I3)
      NPMAX=NP
C
      IF(NQ.LE.NPMAX) GO TO 60
      NPMAX=NQ
 60   IF(NR.LE.NPMAX) GO TO 61
      NPMAX=NR
 61   CONTINUE
      READ(5,2) ZMIN,ZMAX
 2    FORMAT(1X,2F4.0)
      WRITE (*,*) ZMIN,ZMAX
      MIN=ZMIN
      MA  CHECK TO SEE IF WINDS ARE CONSIDERED TO BE HORIZONTAL
C
 58   IF(NCO.GE.0) GO TO 36
      NEG=-1
      NCO=0
      NR=0
      NC(1)=0
C
 36   CONTINUE
      DO 19 J=1,NR
 19   NUMNC=NUMNC+NC(J)
      N=3+NAO+NBO+NCO+2*(NP+NQ+NR+NUMNA+NUMNB+NUMNC)
      WRITE (*,*) "AC(",N,")"
      IF(N-200) 4,4,3000
 3000 WRITE(25,3001) FF,N
 3001 FORMAT(A1////1X,9HEXECUTION      /////
     +1X,8H    ****,3X,16HDIMENSION OF N (,I4,23H ) EXCEEDS THAT ALLOWED
     2////1X,8H    ****,3X,29HPROGRAMME CANNOT BE CONTINUED//)
      PAUSE " MATRIX > 200*200"
      STOP
 4    CONTINUE
      NE=N+1
      N2=2*N
C
C     GET NAME OF SPP INPUT FILE
C
      WRITE (*,*) " "
      WHICH=" FIRST"
      WRITE (*,3002) WHICH
 3002 FORMAT (" ENTER INTEGER PART (XXX) OF ",A6," FILENAME  ",$)
      READ (*,*) IFILE
      INSTART="SPP - GR "
```

```
      I100=IFILE/100
      I10=(IFILE-I100*100)/10
      I1=IFILE-I100*100-I10*10
      DUMSTA(10)=CHAR(I100+48)
      DUMSTA(11)=CHAR(I10+48)
      DUMSTA(12)=CHAR(I1+48)
      WHICH="  LAST"
      WRITE (*,3002) WHICH
      READ (*,*) IEND
      INEND="SPP - GR "
      I100=IEND/100
      I10=(IEND-I100*100)/10
      I1=IEND-I100*100-I10*10
      DUMEND(10)=CHAR(I100+48)
      DUMEND(11)=CHAR(I10+48)
      DUMEND(12)=CHAR(I1+48)
      WRITE (*,*) " "
C
C     ENTER NUMBER OF POINTS PER HEIGHT TO BR PROCESSED
C
      WRITE (*,3003)
 3003 FORMAT (" NUMBER OF POINTS PER HEIGHT?    ",$)
      READ (*,*) NPOINTS
      WRITE (*,*) " "
C
      WRITE (*,3004)
 3004 FORMAT (" PRINT DATA TO SCREEN? Y OR N    ",$)
      READ (*,3005) NPRINT
 3005 FORMAT(A1)
      WRITE (*,*) " "
      RETURN
      END
```

```
      SUBROUTINE MAPSTAR
C                                          SEPTEMBER 18, 1991
C     STARTING FROM HOUR STRTHR, THIS SUBROUTINE
C     READS RADAR FRAME HEADERS AND RECORDS
C     FROM SCATTERING POINT PARAMETER FILES SPP OR XXX
C     RECORDED ON HARD DRIVE.
C
C     THIS SUBROUTINE PROCESSES ALL DATA IN THE SELECTED
C     INPUT FILES, SUBJECT ONLY TO THE SELECTION OF WHICH HEIGHTS,
C     ZENITH ANGLES AND DAY(S) WILL BE PROCESSED, AS DECLARED
C     IN FILES 7DATES, 7DUNK AND 7GRODAT.
C
      SAVE
      DIMENSION P(200),Q(200,200),NA(10),NB(10),NC(10),AC(200),D(200),
     *SIGMA(200),NTIME(24)
      REAL*4 SPBLOCK(10),SPSKIP(10),FLAG
      CHARACTER*1 NGO,YES,NO,NPRINT,RESULT(72),SOURCE(8),
     *CADD(6),FF,DUM(40)
      CHARACTER*40 DUMMY,INSTART,INEND
      INTEGER*1 ICADD(6),ZERO
      INTEGER*2 STRTHR
      INTEGER*4 M,IDATE(3),ISEC1,ISEC2,ISEC3,ISEC4
      INTEGER*4 IFIX,SKIP
      COMMON/HRSTRT/ STRTHR,RNGCOR,ISEC1,ISEC2,ISEC3
      COMMON/WINDS/ N,Q,NOP,ZMIN,MIN,ZMAX,MAX,NA,NB,NC,NAO,NBO,
     *NCO,SUM,AC,NTIME,NP,NQ,NR,RESULT,SOURCE,PERIOD,
     *FF,CADD,Z
      COMMON/ECHOES/ P,VEL,SINJ(10),COSJ(10),DCL,DCM,DCN,SUM1,
     *SUM2,SUM3,D,SIGMA,AZENMIN,AZENMAX,IEND
      COMMON/EXTRAS/ IUNIT,ICADD,NGO,STRTDA,ENDDAY,
     *LEAPYR,JMO,MNO(20,24),NFILE,LENGTH,NEG,NPMAX,ZERO,
     *NPRINT,DAY,JOBAD
      COMMON/GENE/ M,UR,MY,MO,JO,LTIMH,LTIMM,MSEC,EL3,EM3
      COMMON/FSPEC/IFILE,DUMMY,INSTART,INEND,NSTART,NPOINTS
      COMMON/HEIGHTS/NOZ,NQUAD
      COMMON/TEMPO/IDATE,LHOUR,LMIN,LSEC
      EQUIVALENCE(DUMMY,DUM)
      DATA MAPRD,NEXIT,RADIAN/0,0,57.295779/
      IF(MAPRD.EQ.1) GO TO 1
      WRITE (*,*) " "
      WRITE (*,*) " START HOUR ",STRTHR
      MAPRD=1
      UR=1.0
      IZS=0
      TWOPI=6.2831852
      FLAG=-999.0
      ONE=1.0
      THREE=3.0
      LINE=0
      IHOLD=64
      LHOLD=-5
      MHOLD=0
      NZ=0
      M=0
      JSEC=ISEC1
      ISEC3=0
      NSTART=1
```

```
      ZIP=0.0
      DUMMY="SPP - GR "
      I100=IFILE/100
      I10=(IFILE-I100*100)/10
      I1=IFILE-I100*100-I10*10
      DUM(10)=CHAR(I100+48)
      DUM(11)=CHAR(I10+48)
      DUM(12)=CHAR(I1+48)
C
C     OPEN FIRST SPP - GR XXX DISC FILE
C
      OPEN (22,FILE=DUMMY,FORM="UNFORMATTED")
      WRITE (25,1011) DUMMY
C
      WRITE (*,*) " "
      WRITE(*,995) NPOINTS,NQUAD
  995 FORMAT(" PROCESSING",I3," POINT(S) PER 1KM HEIGHT
     * IN QUADRANT",I3/)
      WRITE(*,996) AZENMIN,AZENMAX
  996 FORMAT(" ZENITH ANGLE SPREAD",F4.0," TO ",F4.0," DEG."/)
      ZENMIN=COS(AZENMIN/RADIAN)
      ZENMAX=COS(AZENMAX/RADIAN)
      YES="Y"
      NO="N"
      NFRAME=0
      NULL=0
      NH=(MAX-MIN)/3-1
      WRITE (*,*) " "
      WRITE (*,*) " "
      WRITE (*,*) " ******** EXECUTION PROCEEDING ********"
      WRITE (*,*) " "
C
    1 READ (22,END=5) SPBLOCK
      IF (SPBLOCK(1).NE.FLAG) GO TO 2
      MY=SPBLOCK(2)
      MO=SPBLOCK(3)
      IF(MO.LT.1.OR.MO.GT.12) GO TO 4
      JO=SPBLOCK(4)
      LTIMH=SPBLOCK(5)
      LTIMM=SPBLOCK(6)
      MSEC=SPBLOCK(7)
C
      IF(LTIMH.EQ.LHOLD) GO TO 1
      MH=M-MHOLD
      MHOLD=M
      LHOLD=LTIMH
      LINE=LINE+1
      IF (LINE.LE.48) GO TO 195
      NOP=NOP+1
      LINE=0
      WRITE (26,1003) FF
      WRITE (26,1004) RESULT,SOURCE,NOP
  195 SPBLOCK(4)=JO
      SPBLOCK(5)=LTIMH
      SPBLOCK(6)=LTIMM
      SPBLOCK(7)=MSEC
      WRITE (*,998) IFILE,(SPBLOCK(II),II=2,10),MH
  998 FORMAT(1X,"SPP - GR ",I3,9F8.3,I8)
```

```
      WRITE (26,336) IFILE,(SPBLOCK(II),II=1,11),NH
      GO TO 1
C
C     USE FILE TGRODAT HEIGHT RANGE SPECIFICATION
C
    2 IF(SPBLOCK(1).LT.ZMIN) GO TO 1
      IF(SPBLOCK(1).GT.ZMAX) GO TO 1
C
C     DETERMINE DIRECTION COSINES
C
      EL3=SIN(SPBLOCK(3)/RADIAN)
      EM3=SIN(SPBLOCK(4)/RADIAN)
      EN3=SQRT(ONE-EL3*EL3-EM3*EM3)
C
C     CHECK ZENITH ANGLE
C
      IF(EN3.LT.ZENMAX.OR.EN3.GT.ZENMIN) GO TO 1
C
      Z=SPBLOCK(1)
C
C     ELIMINATE IF LINEARLY POLARIZED
C
      ANGLE=SPBLOCK(6)-SPBLOCK(8)
      IF (ABS(ANGLE).LT.45.0) GO TO 1
      IF (ABS(ANGLE).GT.135.0.AND.ABS(ANGLE).LT.225.0) GO TO 1
      IF (ABS(ANGLE).GT.315.0.AND.ABS(ANGLE).LT.360.0) GO TO 1
C
C     ELIMINATE IF X POLARIZATION
C
      IF (ANGLE.GT.-135.0.AND.ANGLE.LT.-45.0) GO TO 1
      IF (ANGLE.GT.-135.0.AND.ANGLE.LT.-45.0) GO TO 1
      IF (ANGLE.GT.225.0.AND.ANGLE.LT.315.0) GO TO 1
C
C     SELECT QUADRANTS TO BE PROCESSED
C
C     IF NQUAD = O    PROCESS ALL QUADRANTS
C                1            FIRST AND SECOND QUADRANTS
C                2            SECOND AND THIRD QUADRANTS
C                3            THIRD AND FOURTH QUADRANTS
C                4            FOURTH AND FIRST QUADRANTS
C
      IF (NQUAD.EQ.0) GO TO 25
      GO TO (21,22,23,24) NQUAD
   21 IF ((EL3.GT.ZIP.AND.EM3.GT.ZIP).OR.(EL3.GT.ZIP.AND.EM3.LT.ZIP))
     *GO TO 25
      GO TO 1
   22 IF ((EL3.GT.ZIP.AND.EM3.LT.ZIP).OR.(EL3.LT.ZIP.AND.EM3.LT.ZIP))
     *GO TO 25
      GO TO 1
   23 IF ((EL3.LT.ZIP.AND.EM3.LT.ZIP).OR.(EL3.LT.ZIP.AND.EM3.GT.ZIP))
     *GO TO 25
      GO TO 1
   24 IF ((EL3.LT.ZIP.AND.EM3.GT.ZIP).OR.(EL3.GT.ZIP.AND.EM3.GT.ZIP))
     *GO TO 25
      GO TO 1
C
C     SELECT NUMBER OF POINTS AT EACH HEIGHT
C
```

```
   15 IZ=I
      IF(IZ.EQ.IHOLD) IZS=IZS+1
      IF(IZ.NE.IHOLD) IZS=1
      IF(IZ.NE.IHOLD) IHOLD=IZ
      IF(IZS.LE.NPOINTS) GO TO 3
      GO TO 1
C
C     DETERMINE WHETHER POINT FALLS IN DESIRED TIME INTERVAL
C
    3 DAY=FLOAT(MY)*10000.+FLOAT(MO)*100.+FLOAT(JO)
      IF(DAY.LT.STRTDA) GO TO 1
      IF(DAY.LE.ENDDAY) GO TO 30
      IF(DAY.NE.STRTDA.AND.LTIMH.GE.STRTHR) GO TO 6
  600 WRITE (*,*) " LOST FRAME SYNC - LOOK FOR NEXT TIME FRAME"
      WRITE (25,*) " LOST FRAME SYNC - LOOK FOR NEXT TIME FRAME"
  667 READ (22,END=5) SPBLOCK
      IF (SPBLOCK(1).NE.FLAG) GO TO 667
      WRITE (*,*) " FOUND IT!"
      WRITE (25,*) " FOUND IT!"
      GO TO 1
   30 IF(DAY.EQ.STRTDA.AND.LTIMH.LT.STRTHR) GO TO 1
      IF (STRTHR.LT.0.5) GO TO 31
      IF(DAY.NE.STRTDA.AND.LTIMH.GE.STRTHR) GO TO 8
   31 VEL=SPBLOCK(2)
      IF(NFRAME.EQ.1) IHOLD=64
      NFRAME=0
      GO TO 11
    4 NEXIT=NEXIT+1
      WRITE(25,999) DUMMY,UR,MY,MO,JO,LTIMH,LTIMM,MSEC
  999 FORMAT(///1X,A12," READ ERROR AT OR NEAR ECHO NUMBER ",
     *F7.0,6X,6I2)
      WRITE(*,999) DUMMY,UR,MY,MO,JO,LTIMH,LTIMM,MSEC
      IF(NEXIT.LT.1000) GO TO 1
      WRITE (*,1000) DUMMY,MY,MO,JO,LTIMH,LTIMM,MSEC,DUMMY
 1000 FORMAT(///1X,A12," BAD FILE ",3I2,1X,3I2,A12)
      WRITE (25,1000) DUMMY,MY,MO,JO,LTIMH,LTIMM,MSEC,DUMMY
      PAUSE "CR"
      STOP
C
C     SELECT NEXT SPP INPUT FILE
C
    5 CONTINUE
      IF(UR.LT.0.6) GO TO 8
      IFILE=IFILE+1
      CALL IBAD (IFILE,SKIP)
      IF(IFILE.GT.IEND) GO TO 8
      IF (SKIP.EQ.1) GO TO 5
      CLOSE (22)
      I100=IFILE/100
      I10=(IFILE-I100*100)/10
      I1=IFILE-I100*100-I10*10
      DUM(10)=CHAR(I100+48)
      DUM(11)=CHAR(I10+48)
      DUM(12)=CHAR(I1+48)
      UR=1.0
      IHOLD=64
      WRITE (25,1011) DUMMY
 1011 FORMAT(" ACCESSING FILE ",A12)
```

```
      OPEN (22,FILE=DUMMY,FORM="UNFORMATTED")
      GO TO 1
C
C     ACCEPTABLE POINT HAS BEEN FOUND
C
   11 DO 599 I=1,10
      CHCKIP(I)=SPBLOCK(I)
  599 CONTINUE
      M=M+1
      UR=UR+ONE
 1001 FORMAT(I8,$)
      IF(MOD(M,500).NE.0) GO TO 6
      WRITE (*,*) "    ",M," POINTS PROCESSED"
      CALL DateTime
      ISEC4=3600*LHOUR+60*LMIN+LSEC
      IF(ISEC4.GT.JSEC) GO TO 6
      ISEC3=24*3600
      JSEC=20*60
    6 IF(NPRINT.EQ.YES) WRITE (*,1002) UR,MY,MO,JO,LTIMH,LTIMM,Z,VEL,
     *EL3,EM3
 1002 FORMAT(F7.0,I3,I3,I3,I3,I2,2X,2F6.1,2F6.3)
      IF(M.EQ.1) STRTDA=DAY
C
C     CALCULATE TIME WITH RESPECT TO INPUT PERIODICITY
C
      CALL DAYMON (MYRDAY,LEAPYR,MO,JO)
      TMINIT=1440.0*FLOAT(MYRDAY-1)+60.0*FLOAT(LTIMH)+FLOAT(LTIMM)
     *+FLOAT(MSEC)/60.0
      THOUR=TMINIT/60.0
      NEWDAY=THOUR/PERIOD
      T=(THOUR/PERIOD-FLOAT(NEWDAY))*TWOPI
      DO 7 J=1,NPMAX
      FJT=FLOAT(J)*T
      SINJ(J)=SIN(FJT)
      COSJ(J)=COS(FJT)
    7 CONTINUE
C
C     ENTER COUNT IN POINT RATE MATRIX
C
      LT=LTIMH-STRTHR+1
      IF(LT.LE.0) LT=LT+24
      Z0=ZMIN-0.01
      Z3=ZMIN+THREE
      DO 13 I=1,NH
      IF(Z.GT.Z0.AND.Z.LE.Z3) GO TO 14
      Z0=Z0+THREE
      Z3=Z3+THREE
   13 CONTINUE
   14 MNO(I,LT)=MNO(I,LT)+1
      RETURN
C
C     SPP DATA READ AND PROCESSED. FLAG WINDS CALCULATION
C
    8 UR=0.5
      WRITE (26,1003) FF
 1003 FORMAT(A1)
      NOP=NOP+1
      WRITE (26,1004) RESULT,SOURCE,NOP
```

```
1004 FORMAT(  1X,70A1,8A1,16X,"PAGE  ",I3)
     CLOSE (22)
     INEND=DUMMY
     IF (IFILE.GE.IEND) RETURN
     IF (JERR.EQ.16) RETURN
     WRITE (25,1005) DUMMY,IFILE,IEND
1005 FORMAT ( /," FILE ERROR RESULTED IN EXIT FROM ",A11
    *" CURRENT FILE DESIGNATOR IS ",I3,". IEND SET TO ",I[illegible]
     WRITE (*,1005) DUMMY,IFILE,IEND
     WRITE (25,1006) NFRAME,MY,MO,JO,LTIMH,LTIMM,SPSKIP,SPBLOCK
1006 FORMAT (" NFRAME =",I3," TIME ",4I3,I2 "  SPSKIP ",[illegible]
    *" SPBLOCK ",10E10.3 /)
     WRITE (*,1006) NFRAME,MY,MO,JO,LTIMH,LTIMM,SPSKIP,SPBLOCK
     WRITE (*,*) " ATTEMPTING WIND ANALYSIS"
     RETURN
     END
```

```
      SUBROUTINE MATSIN(A,N,DETERM)
C                                              MAY 3, [illegible]
C     MATRIX INVERSION, TO 200 X 200
C     (SIZE OF MATRIX DETERMINED BY N)
C
      DIMENSION IPIVOT(200),A(200,200),INDEX(200,2),PIVOT(200)
      EQUIVALENCE (IROW,JROW),(ICOLUM,JCOLUM),(AMAX,SWAP)
C
C     INITIALIZATION
C
      ZERO=0.0
      DETERM=0.0
 15   DO 20 J=1,N
 20   IPIVOT(J)=0
C
C     SEARCH FOR PIVOT ELEMENT
C
 30   DO 550 I=1,N
 40   AMAX=0.0
 45   DO 105 J=1,N
 50   IF(IPIVOT(J)-1) 60,105,60
 60   DO 100 K=1,N
 70   IF(IPIVOT(K)-1) 80,100,740
 80   IF(ABS(AMAX)-ABS(A(J,K))) 85,100,100
 85   IROW=J
 90   ICOLUM=K
 95   AMAX=A(J,K)
 100  CONTINUE
 105  CONTINUE
 110  IPIVOT(ICOLUM)=IPIVOT(ICOLUM)+1
C
C     INTERCHANGE ROWS TO PUT PIVOT ELEMENT ON DIAGONAL
C
 130  IF(IROW-ICOLUM) 140,260,140
 140  DETERM=-DETERM
 150  DO 200 L=1,N
 160  SWAP=A(IROW,L)
 170  A(IROW,L)=A(ICOLUM,L)
 200  A(ICOLUM,L)=SWAP
 260  INDEX(I,1)=IROW
 270  INDEX(I,2)=ICOLUM
 310  PIVOT(I)=A(ICOLUM,ICOLUM)
      ABSPIV=ABS(PIVOT(I))
      IF(ABSPIV.GT.ZERO) GO TO 320
      DETERM=-13.0
      RETURN
 320  DETERM=DETERM+ALOG10(ABSPIV)
      IF(DETERM.GT.-12.0) GO TO 330
      RETURN
C
C     DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
 330  A(ICOLUM,ICOLUM)=1.0
 340  DO 350 L=1,N
 350  A(ICOLUM,L)=A(ICOLUM,L)/PIVOT(I)
```

```
C
C     REDUCE NON-PIVOT ROWS
C
 380  DO 550 L1=1,N
 390  IF(L1-ICOLUM) 400,550,400
 400  T=A(L1,ICOLUM)
 420  A(L1,ICOLUM)=0.0
 430  DO 450 L=1,N
 450  A(L1,L)=A(L1,L)-A(ICOLUM,L)*T
 550  CONTINUE
C
C     INTERCHANGE  COLUMNS
C
 600  DO 710 I=1,N
 610  L=N+1-I
 620  IF(INDEX(L,1)-INDEX(L,2)) 630,710,630
 630  JROW=INDEX(L,1)
 640  JCOLUM=INDEX(L,2)
 650  DO 705 K=1,N
 660  SWAP=A(K,JROW)
 670  A(K,JROW)=A(K,JCOLUM)
 700  A(K,JCOLUM)=SWAP
 705  CONTINUE
 710  CONTINUE
 740  RETURN
      END




      SUBROUTINE SDESIG (NAME,A)
C                                                        MARCH 30, 1990
C     ADDS DESIGNATOR "A" TO FILENAME "NAME"
C     AS PREFIX
C
      CHARACTER*1 A(6),NAME(40)
C
      DO 1 I=40,7,-1
      NAME(I)=NAME(I-6)
    1 CONTINUE
    2 DO 3 J=1,6
      NAME(J) = A(J)
    3 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SDIANA
C                                                    JUNE 1[illegible], 1[illegible]
C     PRINTS OUT HOUR BY HOUR WIND PROFILES OF FUNDAMENTAL [illegible]
HOURS
C
C
      DIMENSION Q([illegible],[illegible]),NA([illegible]),NB([illegible]),NC([illegible]),
     1AC([illegible]),U(24),NTIME(24),NANEW([illegible])
      CHARACTER*1 RESULT(72),SOURCE(8),FF,CADD(6)
      CHARACTER*40 ARCHIT
      INTEGER*1 ICADD(6)
      INTEGER*2 STRTHR
      COMMON/HRSTRT/ STRTHR
      COMMON/WINDS/ N,Q,NOP,ZMIN,MIN,ZMAX,MAX,NA,NB,NC,NA[illegible],NB[illegible],
     *NCO,SUM,AC,NTIME,NP,NQ,NR,RESULT,SOURCE,PERIOD,
     *FF,CADD,Z
      COMMON/EXTRAS/ IUNIT,ICADD,NGO

C
      CENTIM=1.0
      ARCHIT="ARCHIT "
      CALL SDESIG (ARCHIT,CADD)
      OPEN (14,FILE=ARCHIT,FORM="UNFORMATTED")
C
      NOP=NOP+1
      WRITE (26,998) FF,RESULT,SOURCE,NOP
 998  FORMAT(A1/72A1,8A1,26X,"PAGE",I3/)
C
C     EAST-WEST WIND COMPONENT, HOUR BY HOUR
C
      WRITE(26,597)MIN,MAX
 597  FORMAT(1X,52HEAST-WEST COMPONENTS OF THE MEAN WIND
     *, HOUR BY HOUR,
     1/1X,34HAS DETERMINED FOR THE HEIGHT RANGE,I5,6H KM TO,I5,4H KM.
     2/)
      WRITE(26,600)NTIME
 600  FORMAT(1X,6HHEIGHT,24I5/1X)
      DO 128 J=1,NP
 128  NANEW(J)=NA(J)
      NPNEW=NP
      KEND=0
      NAOE=NAO+1
      NBOE=NBO+1
      NCOE=NCO+1
      NSIGN=-1
 105  KA=KEND
      DO 307 KZ=MIN,MAX
      UO=0.0
      DO 303 LT=1,24
 303  U(LT)=0.0
      Z=MAX+MIN-KZ
      NZ=Z
      S=(2.0*Z-ZMAX-ZMIN)/(ZMAX-ZMIN)+1.0E-6
      DO 304 K=1,NAOE
      KUA=K+KA
 304  UO=UO+AC(KUA)*S**(K-1)
      UO=UO*CENTIM
      IF(NPNEW) 310,310,308
```

```
 308  DO 306 IT=1,24
      KEND=KA+NAOE
      LT=IT+STRTHR-1
      IF(LT.GE.24) LT=LT-24
      DO 305 J=1,NPNEW
      KSTART=KEND
      NAEND=NANEW(J)+1
      KEND=KSTART+NAEND
      JLT=J*LT
      DO 305 K=1,NAEND
      KU=K+KSTART
 305  U(IT)=U(IT)+AC(KU)*S**(K-1)*SIN(FLOAT(JLT)*0.2618)
      DO 312 J=1,NPNEW
      KSTART=KEND
      NAEND=NANEW(J)-1
      KEND=KSTART+NAEND
      JLT=J*LT
      DO 312 K=1,NAEND
      KU=K+KSTART
 312  U(IT)=U(IT)+AC(KU)*S**(K-1)*COS(FLOAT(JLT)*0.2618)
      U(IT)=U(IT)*CENTIM+UO
      IF(ABS(U(IT))-999.0) 306,306,309
 309  U(IT)=SIGN(999.0,U(IT))
 306  CONTINUE
      GO TO 313
 310  KEND=KA+NAOE
      U(1)=UO
      IF(ABS(U(1))-999.0) 107,107,311
 311  U(1)=SIGN(999.0,U(1))
 107  DO 108 LT=2,24
 108  U(LT)=U(1)
  313 WRITE(14) NZ,U,U(1)
  307 WRITE(26,888) NZ,U
 888  FORMAT(1X,I4,3X,24F5.0)
      IF(NSIGN) 129,131,133
C
C     NORTH-SOUTH WIND COMPONENTS, HOUR BY HOUR.
C
 129  NAOE=NBOE
      NPNEW=NQ
      DO 130 J=1,NQ
 130  NANEW(J)=NB(J)
      NSIGN=0
      NOP=NOP+1
      WRITE(26,998) FF, RESULT,SOURCE,NOP
      WRITE(26,598)MIN,MAX
 598  FORMAT(1X,54HNORTH-SOUTH COMPONENTS OF THE MEAN WIND,
     * HOUR BY HOUR.
     1/1X,34HAS DETERMINED FOR THE HEIGHT RANGE,I5,6H KM TO,I5,4H KM.
     2/)
      WRITE(26,600) NTIME
      GO TO 105
C
C     VERTICAL WIND COMPONENTS, HOUR BY HOUR.
C
 131  NAOE=NCOE
      CENTIM=100.0
      NPNEW=NP
```

```
      DO 132 J=1,NR
132   NANEW(J)=NC(J)
      NSIGN=2
      NOP=NOP-1
      WRITE(26,998) FF, RESULT,SOURCE,NOP
      WRITE(26,599)MIN,MAX
599   FORMAT(1X,50HVERTICAL COMPONENT OF THE MEAN WIND,
     * HOUR BY HOUR,
     1/1X,34HAS DETERMINED FOR THE HEIGHT RANGE,I5,6H KM TO,I5,3H KM.
     2/"  **** NOTE THAT VERTICAL WINDS ARE IN CENTIMETERS PER SECOND!
     3 ****"/)
      WRITE(26,600) NTIME
      GO TO 105
133   CLOSE (14)
      RETURN
      END
```

```
      SUBROUTINE SVARY
C                                                  MAY 4,19[illegible]
C     CALCULATES THE AMPLITUDE AND PHASE OF PREVAILING AND PERIODIC
C     COMPONENTS, UP TO THE FOURTH HARMONIC,
C     TOGETHER WITH THE MOST PROBABLE ERROR IN EACH.
C     PRINTS THESE OUT AT 1 KM INTERVALS OVER THE HEIGHT RANGE
SPECIFIED.
C
      DIMENSION Q(200,200),NA(10),NB(10),NC(10),
     1NTIME(24),AC(200),SI(10),CO(10),AU(10),PH(10),SIGSIN(10),
     2SIGSC(10),SIGCOS(10),SIGPH(10),SIGAMP(10),ERPH(10),ERAMP(10),
     3PTEM(4)
      CHARACTER*1 RESULT(72),SOURCE(8),FB,FF,CADD(6),NGO
      CHARACTER*40 TIDE,ERROR,ATIDE
      COMMON/WINDS/ N,Q,NOP,ZMIN,MIN,ZMAX,MAX,NA,NB,NC,NAO,NBO,NCO,SUM,
     1AC,NTIME,NP,NQ,NR,RESULT,SOURCE,PERIOD,FF,CADD,Z,A(200,200)
      COMMON/EXTRAS/ IUNIT,ICADD,NGO,STRTDA,ENDDAY,LEAPYR,JMC,
     *MNO(20,24),NFILE,LENGTH,NEG,NPMAX,ZERO,NPRINT
C
      CENTIM=1.0
      TIDE="TIDE "
      CALL SDESIG (TIDE,CADD)
      OPEN (15,FILE=TIDE,FORM="UNFORMATTED")
      ERROR="ERROR "
      CALL SDESIG (ERROR,CADD)
      OPEN (16,FILE=ERROR,FORM="UNFORMATTED")
      ATIDE="ATIDE "
      CALL SDESIG (ATIDE,CADD)
      OPEN (17,FILE=ATIDE,FORM="FORMATTED")
C
      FB=" "
      NOP=NOP+1
      WRITE(26,998) FF,RESULT,SOURCE,NOP
      WRITE(17,998) FB,RESULT,SOURCE,NOP
 998  FORMAT(A1/72A1,8A1,26X,"PAGE",I3/)
      NAOE=NAO+1
      NBOE=NBO+1
      NCOE=NCO+1
      KEND=0
      DO 86 J=1,4
      PTEM(J)=PERIOD/FLOAT(J)
 86   CONTINUE
      NSIGN=-1
      WRITE(26,597) MIN,MAX
      WRITE(17,597) MIN,MAX
 597  FORMAT(1X58H EAST-WEST COMPONENTS OF THE MEAN WIND, AMPLITUDE AND
     1PHASE,/1X,35H AS DETERMINED FOR THE HEIGHT RANGE,I5,6H KM TO,I5,
     24H KM.//)
 98   IF(NP) 89,89,88
 89   WRITE(26,87)
      WRITE(17,87)
 87   FORMAT(1X,18HHEIGHT MEAN ERROR/1X)
      GO TO 105
 88   GO TO (99,99,101,103),NP
 99   WRITE(26,100) (PTEM(J),J=1,2)
      WRITE(17,100) (PTEM(J),J=1,2)
 100  FORMAT(22X,F6.1,15H HOUR COMPONENT,3X,F9.1,15H HOUR COMPONENT/1X,
     175HHEIGHT   MEAN    ERROR  AMP    ERROR PHASE    ERROR   AMP    ERROR P
```

```
    2HASE   ERROR/1X)
     GO TO 105
101  WRITE(26,102) (PTEM(J),J=1,3)
     WRITE(17,102) (PTEM(J),J=1,3)
102  FORMAT(22X,F6.1,15H HOUR COMPONENT,3X,F9.1,15H HOUR COMPONENT,3X,
    1F9.1,15H HOUR COMPONENT,           1X,1
    222HHEIGHT  MEAN    ERROR  AMP   ERROR PHASE   ERROR  AMP   ERROR P
    PHASE   ERROR  AMP   ERROR PHASE   ERROR 1X)
     GO TO 105
103  WRITE(26,104) (PTEM(J),J=1,4)
     WRITE(17,104) (PTEM(J),J=1,4)
104  FORMAT(22X,F6.1,15H HOUR COMPONENT,3X,F9.1,15H HOUR COMPONENT,3X,
    1F9.1,15H HOUR COMPONENT,3X,F9.1,15H HOUR COMPONENT 1X,          1
    229HHEIGHT  MEAN    ERROR  AMP   ERROR PHASE   ERROR  AMP   ERROR P
    3HASE   ERROR  AMP   ERROR PHASE   ERROR  AMP   ERROR PHASE   ERROR
    4/1X)
105  KA=KEND
     DO 128 KZ=MIN,MAX
     UO=0.0
     Z=MAX+MIN-KZ
     NZ=Z
     S=(2.0*Z-ZMAX-ZMIN)/(ZMAX-ZMIN)+1.0E-06
     DO 106 K=1,NAOE
     KUA=K+KA
106  UO=UO+AC(KUA)*S**(K-1)
     UO=UO*CENTIM
     SIGUO=0.0
     DO 107 K=1,NAOE
     DO 107 L=1,NAOE
     KS=K+KA
     LS=L+KA
107  SIGUO=SIGUO+S**(K-1)*S**(L-1)*A(KS,LS)*SUM
     ASIGUO=ABS(SIGUO)+0.01
     SIGN=SIGUO/ASIGUO
     EO=SIGN*SQRT(ASIGUO)*CENTIM
     IF(NP) 126,126,108
108  KEND=KA+NAOE
     NUMNA=0
     DO 109 J=1,NP
     SI(J)=0.0
     CO(J)=0.0
     SIGSIN(J)=0.0
     SIGSC(J)=0.0
     SIGCOS(J)=0.0
     NUMNA=NUMNA+NA(J)
109  CONTINUE
     DO 121 J=1,NP
     NUSIN=NP+NUMNA
     KSTART=KEND
     NAEND=NA(J)+1
     KEND=KSTART+NAEND
     DO 110 K=1,NAEND
     KS=K+KSTART
     SI(J)=SI(J)+AC(KS)*S**(K-1)
     KC=KS+NUSIN
110  CO(J)=CO(J)+AC(KC)*S**(K-1)
     SINSQJ=SI(J)**2
     COSSQJ=CO(J)**2
```

```
      SUMSQJ=SINSQJ+COSSQJ
      AU(J)=SQRT(SUMSQJ)*CENTIM
      FJ=J
      FJ=PERIOD/FJ
      IF(CO(J)) 114,111,115
111   IF(SI(J)) 113,113,112
112   PH(J)=FJ/4.0
      GO TO 116
113   PH(J)=FJ*0.75
      GO TO 116
114   PH(J)=FJ*0.5+(FJ/6.28318)*ATAN(SI(J)/CO(J))
      GO TO 116
115   PH(J)=(FJ/6.28318)*ATAN(SI(J)/CO(J))
116   IF(PH(J)) 117,118,118
117   PH(J)=FJ+PH(J)
118   DO 119 K=1,NAEND
      DO 119 L=1,NAEND
      KS=K+KSTART
      LS=L+KSTART
      KC=KS+NUSIN
      LC=LS+NUSIN
      SIGSIN(J)=SIGSIN(J)+S**(K-1)*S**(L-1)*A(KS,LS)
      SIGSC(J)=SIGSC(J)+S**(K-1)*S**(L-1)*A(KS,LC)
119   SIGCOS(J)=SIGCOS(J)+S**(K-1)*S**(L-1)*A(KC,LC)
      PROD=2.0*SI(J)*CO(J)*SIGSC(J)
      SIGPH(J)=(COSSQJ*SIGSIN(J)+SINSQJ*SIGCOS(J)-PROD)*SUM/SUMSQJ**2
      SIGAMP(J)=(SINSQJ*SIGSIN(J)+COSSQJ*SIGCOS(J)+PROD)*SUM/SUMSQJ
      ERPH(J)=SQRT(SIGPH(J))*FJ/6.28318
121   ERAMP(J)=SQRT(SIGAMP(J))*CENTIM
      KEND=KEND+NUSIN
      GO TO (122,122,124,125),NP
122   WRITE(26,123) NZ,UO,EO,(AU(J),ERAMP(J),PH(J),ERPH(J),J=1,2)
      WRITE(17,123) NZ,UO,EO,(AU(J),ERAMP(J),PH(J),ERPH(J),J=1,2)
123   FORMAT(1X,I4,2X,F6.0,2X,F6.0,4(1X,2F6.0,2F7.1))
      WRITE (15) UO,(AU(J),PH(J),J=1,2)
      WRITE (16) EO,(ERAMP(J),ERPH(J),J=1,2)
      GO TO 28
124   WRITE(26,123) NZ,UO,EO,(AU(J),ERAMP(J),PH(J),ERPH(J),J=1,3)
      WRITE(17,123) NZ,UO,EO,(AU(J),ERAMP(J),PH(J),ERPH(J),J=1,3)
      WRITE (15) UO,(AU(J),PH(J),J=1,3)
      WRITE (16) EO,(ERAMP(J),ERPH(J),J=1,3)
      GO TO 28
125   WRITE(26,123) NZ,UO,EO,(AU(J),ERAMP(J),PH(J),ERPH(J),J=1,4)
      WRITE(17,123) NZ,UO,EO,(AU(J),ERAMP(J),PH(J),ERPH(J),J=1,4)
      WRITE (15) UO,(AU(J),PH(J),J=1,4)
      WRITE (16) EO,(ERAMP(J),ERPH(J),J=1,4)
      GO TO 28
126   WRITE(26,123) NZ,UO,EO
      WRITE(17,123) NZ,UO,EO
      WRITE (15) UO
      WRITE (16) EO
      KEND=KA+NAOE
 28   CONTINUE
128   CONTINUE
      IF(NSIGN) 129,131,133
129   NAOE=NBOE
      NP=NQ
      DO 130 J=1,NQ
```

```
130  NA(J)=NB(J)
     NSIGN=0
     NOP=NOP+1
     WRITE(26,998) FF,RESULT,SOURCE,NOP
     WRITE(26,598) MIN,MAX
     WRITE(17,998) FF,RESULT,SOURCE,NOP
     WRITE(17,598) MIN,MAX
598  FORMAT(1X," NORTH-SOUTH COMPONENTS OF THE MEAN WIND, AMPLITUDE AN
    1D PHASE"/1X," AS DETERMINED FOR THE HEIGHT RANGE",I5,6H KM TO,I5,
    2" KM."//)
     GO TO 98
131  IF(NEG.EQ.-1) GO TO 133
     NAOE=NCOE
     NP=NR
     DO 132 J=1,NR
132  NA(J)=NC(J)
     NSIGN=1
     NOP=NOP+1
     WRITE(26,998) FF, RESULT,SOURCE,NOP
     WRITE(26,599) MIN,MAX
     WRITE(17,998) FF, RESULT,SOURCE,NOP
     WRITE(17,599) MIN,MAX
599  FORMAT(1X," VERTICAL   COMPONENTS OF THE MEAN WIND, AMPLITUDE AND
    1 PHASE"/1X," AS DETERMINED FOR THE HEIGHT RANGE",I5,6H KM TO,I5,
    2" KM."/"  **** NOTE THAT VERTICAL WINDS ARE IN CENTIMETERS
    3 PER SECOND! ****"//)
     CENTIM=100.0
     GO TO 98
133  WRITE(26,85) FF
     WRITE(17,85) FF
85   FORMAT(A1)
     CLOSE (17)
     RETURN
     END
```

```
      SUBROUTINE TRANS7 (INFILE,OUTFILE)
C                                                SEPTEMBER 17, 1992
C     CONVERTS FILE INFILE TO FILE OUTFILE
C     IN A FORMAT SUITABLE FOR PLOTTING BY
C     SPYGLASS TRANSFORM.
C
C     FOR USE WITH TRANCFORM 2.1
C
      SAVE
      CHARACTER*1 TAB,FF,REFMT(250),RESULT(72),SOURCE(8)
      CHARACTER*4 CADD
      CHARACTER*40 INFILE,OUTFILE
      COMMON/WINDS/ N,Q(200,200),NOP,ZMIN,MIN,ZMAX,MAX,
     *NA(10),NB(10),NC(10),NAO,NBO,NCO,SUM,AC(200),NTIME(24),
     *NP,NQ,NR,RESULT,SOURCE,PERIOD,FF,CADD,Z
      COMMON/HEIGHTS/NOZ
      DATA LOOP/0/
C
C     OPEN I/O FILES
C
      OPEN(15,FILE=INFILE,FORM="UNFORMATTED")
      CALL SDESIG (OUTFILE,CADD)
      OPEN(16,FILE=OUTFILE,FORM="FORMATTED")
      IF(LOOP.EQ.1) GO TO 1
      LOOP=1
      ISTOP=0
      TAB=CHAR(9)
      NCHAR=232
    1 DO 8 I=1,NOZ,2
      READ (15) (REFMT(IC),IC=1,NCHAR)
      NTOP=NCHAR
      IF=6
    4 IF=IF+1
      IF(IF.EQ.ISTOP) GO TO 7
    5 IF(REFMT(IF).NE.TAB) GO TO 4
      KTOP=NTOP-IF
      DO 6 KK=1,KTOP
      REFMT(IF-1+KK)=REFMT(IF+KK)
    6 CONTINUE
      NTOP=NTOP-1
      ISTOP=NTOP-1
      GO TO 5
    7 LCHAR=NTOP-1
      WRITE (16,100) (REFMT(IC),IC=8,LCHAR)
  100 FORMAT(250A1)
    8 CONTINUE
      CLOSE (15,STATUS="DELETE")
      CLOSE (16)
      RETURN
      END
```

## The IDI wind analysis program IDIWIND.f

This program is a variant of Wind.for, and reads the scattering point parameter files SPP - GR XXX files from the hard drive. Output profiles of zonal, meridional and vertical winds is located in folder OUTFILE as file YYYYYYYY.MAW, where YYYYYYYYY is two digit (leading zero) month, day, hour and minute of the midpoint of the selected interval, whose duration is entered (in minutes) at program prompt.

INPUT

Reads file SET.TIME, which is simply a listing of all the interval center point times to be reduced, formatted as follows

```
8905031215
8905031322
..........
..........
..........
0000000000         ! EOF FLAG
```

Requests length of interval (in minutes)

Requests selection of polarization of received returns

[Winds will be contaminated by using other than the transmitted polarization - O (ordinary) in the case of the AIDA data. L (linear) has been identified, in the AIDA data, as RF interference from an harmonic of a broadcast band AM signal. The origin of X (extraordinary) has not been determined]

OUTPUT

In addition to data file XXXXXXXX.MAW, a status file WIND.TXT contains run diagnostics.

```
C      PROGRAM IDIWIND
*******************************************************************
*                                                                 *
*      IDI WIND-CALCULATION PROGRAM; MAPSTAR RADAR                *
*      COPYRIGHT 1993, HOLODYNE LIMITED 1994.                     *
*      ALL RIGHTS RESERVED.                                       *
*                                                                 *
*******************************************************************
C   MAC VERSION 1.00
C                                  APRIL 15, 1994.
C   THIS PROGRAM WILL CALCULATE ZONAL, MERIDIONAL AND VERTICAL WIND
C   PROFILES IN 1-KM STEPS, WITH  SMOOTHING, FROM REGULAR SCATTERING-
C   POINT PARAMETER FILES, TYPE SPP - OR XXX (ON DISC)
C
C   CURRENTLY SET UP FOR 69-111KM (15 HEIGHTS); NEED TO CHANGE IH
C   DIMENSION OF SPPZ(IH,I,J), AND SOME CODE, TO MATCH OTHER HEIGHT
C   RANGES.
C   THE SCATTERING-POINT PARAMETERS ARE :
C   1.  ALTITUDE (KM).
C   2.  RADIAL VELOCITY (M/SEC).
C   3.  ZENITH ANGLE IN EAST-WEST MERIDIAN (DEGREES).
C   4.  ZENITH ANGLE IN NORTH-SOUTH MERIDIAN (DEGREES).
C   5.  VOLTAGE AMPLITUDE ON #1 DIPOLES.
C   6.  PHASE OF #1 DIPOLES (DEGREES).
C   7.  VOLTAGE AMPLITUDE ON #2 DIPOLES.
C   8.  PHASE OF #2 DIPOLES (DEGREES).
C   9.  ERROR IN 3
C  10.  ERROR IN 4
C  EXPLANATION OF EASILY-REPROGRAMMED PARAMETERS (JUST CHANGE THE SOURCE
C   CODE VALUE GIVEN BELOW:
C    VMAX IS THE LARGEST ALLOWED HORIZONTAL VELOCITY. WE TEST EACH POINT
C   AGAINST VMAX BY PROJECTING ITS RADIAL VELOCITY INTO THE HORIZONTAL
C   PLANE, AND REJECT IT IF IT'S BIGGER THAN VMAX.
C     THMAX IS THE LARGEST ACCEPTABLE RADIAL ZENITH ANGLE.
C     THMIN IS THE SMALLEST ACCEPTABLE RADIAL ZENITH ANGLE.
C     MINH, MINV ARE THE MINIMUM NUMBER OF POINTS.  IF THERE ARE NOT
C   SUFFICIENT POINTS, THAT ALTITUDE IS SKIPPED.
C    NSIGMA IS THE MAXIMUM NUMBER OF STANDARD DEVIATIONS FROM THE FIT AN
C   INDIVIDUAL POINT CAN LIE WITHOUT BEING REJECTED FROM THE VELOCITY
C   CALCULATION.
C     ZMIN IS THE BOTTOM ALTITUDE FOR WHICH WINDS ARE TO BE CALCULATED.
C     ZMAX IS THE TOP ALTITUDE FOR WHICH WINDS ARE TO BE CALCULATED.
C  WIND CALLS INNAME, OUTNAME, WFV, WFH, PHFIT AND SORT.
      REAL*4 PI,VMAX,THMAXV,U(50),V(50),W(50),TRP(50),SUCKS(8),
     1       LINE(10),RMSDVR(50),COSL(2300),COSM(2300),COSN(2300),
     2       DVR(2300),SLOPE,INTERCEPT,VRAD(17)
      INTEGER*4 REJ(4),IH,PARAMETER,TESTFLAG,POINT,NPROFS,NHITES,
     1          NPOINTS(50),INTERVAL,BIGTIME,NPV,NPV0,FITFLAG,MISS,NBAD,
     2          YEAR,MONTH,DAY,HOUR,MINUTE,MINH,MINV,MSEC,IO,NGO,NFILE,
     3          NUMRAD(17),MY,MO,JO,LTIMH,LTIMM,INTHALF,NOWSTART,NOWEND
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      CHARACTER*6 STATE
      CHARACTER*1 ANS1,POLAR
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,U,V,W,TRP,REJ,LINE,WIDTH(50),IWT(2300),
     4           RMSDVR(50),COSL,COSM,COSN,DVR,NUMRAD,VRAD
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1           NSIGMA,TESTFLAG,IH,NPOINTS,INTERVAL,INFILE,OUTFILE,
     2           INPATH,OUTPATH,NPH,NPV,NPV0,
     3           SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,YEAR,MONTH,DAY,HOUR,MINUTE
```

```
C
      PI = 3.14159265
      NTOTAL=0
      VFMAX = 60
      VMAX = 200
      THMAXH = 16
      THMINH = 3
      THMAXV = 5
      THMINV = 0
      THMIN = [illegible]
      MINH = 5
      MINV = 5
      NSIGMA = 3.0
      DO 20000 I=1,8
      SUCKS(I)=999.0
20000 CONTINUE
C
************************************************************************
*   COMMUNICATE WITH USER
************************************************************************
20001 WRITE (*,*) ' HAVE YOU UPDATED FILE SET.TIME? Y OR N'
      READ (*,'(A)') ANS1
      IF (ANS1 .NE. 'Y') THEN
      WRITE (*,*) ' EXITING SO THAT SET.TIME CAN BE UPDATED'
      PAUSE 'CR TO EXIT'
      STOP
      END IF
C
C     OPEN STATS FILE 'WIND.TXT'
C
  665 OPEN (15,FILE='WIND.TXT',ERR=666,STATUS="NEW",IOSTAT=IO,
     1FORM="FORMATTED")
  666 IF (IO.NE.0) THEN
      WRITE (*,*) ' A WIND.TXT FILE ALREADY EXISTS,'
      WRITE (*,*) ' DO YOU WISH TO WRITE OVER IT? Y OR N'
      READ (*,*) ANS1
      IF (ANS1.EQ.'Y') THEN
      OPEN (15,FILE='WIND.TXT',ERR=666,IOSTAT=IO,FORM="FORMATTED")
      CLOSE (15,STATUS='DELETE')
      GO TO 665
      ELSE
      NERR=0
      GO TO 90909
      END IF
      END IF
  667 WRITE (*,*) ' ENTER LENGTH OF INTERVAL (MINUTES)'
      READ (*,*) INTERVAL
      INTHALF=INTERVAL/2
      WRITE (*,*) " SELECT POLARIZATION - ENTER O, X, L OR ALL"
      READ (*,*) POLAR
      INPATH = 'MAXTOR600:INFILES:SPP - GR '
      OUTPATH = 'MAXTOR600:OUTFILES:'
      WRITE (*,*) ' ENTER FIRST SPP - GR  FILE NUMBER'
  668 READ (*,*) NFILE
      LOOP=0
C
C     OPEN 'MIDPOIN TIME OF EACH DATA INTERVAL FILE' = SET.TIME
C
      OPEN (17,FILE="SET.TIME",FORM="FORMATTED")
C
  669 IFILE=NFILE-1
C
C     PROGRAM ACCEPTS SPP DATA OVER 3KM HEIGHT RANGE FOR EACH ALTITUDE
C     AND LOOPS THREE TIMES THROUGH SPP DATA TO PRODUCE OUTPUT AT 1KM
```

```
C     HEIGHT INTERVALS. ZMIN, ZMAX ARE ADJUSTED [illegible]
C
      LOOP=LOOP+1
      IF (LOOP.EQ.1) THEN
      STATE="REWIND"
      ZMIN = 67.5
      ZMAX = 110.5
      END IF
      IF (LOOP.EQ.2) THEN
      STATE="APPEND"
      REWIND (17)
      ZMIN = 68.5
      ZMAX = 110.5
      END IF
      IF (LOOP.EQ.3) THEN
      STATE="APPEND"
      REWIND (17)
      ZMIN = 69.5
      ZMAX = 111.5
      END IF
      NHITES=(ZMAX-ZMIN)/3.0+0.1
      NGO=0
      GO TO 203
*********************************************************************
*     RETURN HERE FOR NEW INPUT FILE
*********************************************************************
20203 NGO=1
  203 CALL INNAME
      WRITE (*,*) 'INFILE = ',INFILE
      NERR=1
      OPEN (18,ERR=90909,FILE=INFILE,STATUS='OLD',IOSTAT=IO,
     *FORM='UNFORMATTED')
      WRITE (15,'(A)') INFILE
 2010 READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .GT. -990.0) GO TO 2010
      WRITE (*,100) (LINE(KK),KK=1,10)
  100 FORMAT (10F8.0)
      WRITE (15,100) (LINE(KK),KK=1,10)
      MY=LINE(2)
      MO=LINE(3)
      JO=LINE(4)
      LTIMH=LINE(5)
      LTIMM=LINE(6)
      MSEC=LINE(7)
      NOWTIME= LTIMM+LTIMH*60+JO*24*60+MO*30*24*60
      REWIND (18)
      IF (NGO.EQ.1) GO TO 20103
*********************************************************************
*    RETURN TO HERE FOR NEW OUTPUT FILE
*********************************************************************
20101 NERR=2
      READ (17,101,END=90910) YEAR,MONTH,DAY,HOUR,MINUTE
  101 FORMAT (5I2)
      IF (MONTH.EQ.0) GO TO 90910
      NPROFS=0
      BIGTIME = MINUTE+HOUR*60+DAY*24*60+MONTH*30*24*60
      NOWSTART=BIGTIME-INTHALF
      IF (NGO.EQ.1) GO TO 670
      IF (NOWTIME.GT.NOWSTART) THEN
      WRITE (*,*) " BAD CHOICE OF SPP INPUT FILE; RE-ENTER SPP INPUT
     *FILENAME"
      CLOSE (18)
      GO TO 668
```

```
      END IF
[illegible] [illegible] WEND=[illegible]TIME+INTHALF
      CALL [illegible]NAME
      NERR=[illegible]
      WRITE (*,*) [illegible]
      WRITE (*,*) [illegible]ILP [illegible]TFILE
      [illegible] [illegible]LE=[illegible]TFILE [illegible]
     *F[illegible]RM=[illegible]F[illegible]RMATTED[illegible]
      WRITE [illegible] [illegible]TFILE
[illegible] [illegible] [illegible]
      [illegible]
      [illegible] IH[illegible] [illegible]
      [illegible]
      W[illegible]IH[illegible] [illegible]
      TRF(IH)=[illegible]
      RMDIVF(IH)=[illegible]
[illegible] CONTINUE
      IH=[illegible]
      MI[illegible]=[illegible]
      NBAD=[illegible]
      D[illegible] [illegible] IREJ=1,4
      REJ(IREJ) = [illegible]
[illegible] CONTINUE
[illegible] WRITE (*,[illegible])
[illegible] FORMAT
     1 (1X,' ALT      U         V       W    TRF  [illegible]  [illegible]  [illegible]  RATE'
     2 '   SLOPE INTERCEPT')
[illegible] NERR=4
      READ (18,ERR=[illegible],IOSTAT=I[illegible],END=[illegible]) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .GT. -[illegible]) GO TO [illegible]
*******************************************************************
*  RETURN TO HERE FOR NEW PROFILE .
*******************************************************************
20133 MY=LINE(2)
      MO=LINE(3)
      JO=LINE(4)
      LTIMH=LINE(5)
      LTIMM=LINE(6)
      MSEC=LINE(7)
      NOWTIME= LTIMM+LTIMH*60+JO*24*60+MO*30*24*60
      IF (NOWTIME.LT.NOWSTART) GO TO 20133
      IF (NOWTIME.GT.NOWEND) THEN
      BACKSPACE (18)
      GO TO 20204
      END IF
      NPROFS=NPROFS+1
      NERR=5
      READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .LT. -990.0) THEN
      NPROFS=NPROFS-1
      GO TO 20133
      END IF
C
C     TEST THE POINT FOR:  ALTITUDE
C
20202 IF (LINE(1) .LT. ZMIN) THEN
      NERR=6
      READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .LT. -990.0) GO TO 20133
      GO TO 20202
      ENDIF
```

```
      IF (LINE(1).GT.DMAX) GO TO [illegible]
      INDEX=(LINE(1)-DMIN)/[illegible]+1
      IF (NFINT(INDEX).GT.[illegible]) GO TO [illegible]
C
C     TEST FOR:
C                      1. PERPENDICULAR HORIZONTAL VELOCITY > VHMAX
C                      2. RADIAL VELOCITY [illegible]
C                      3. RADIAL VELOCITY > VRMAX
C                      4. POLARIZATION
C
C
      TESTFLAG = 1
      [illegible]
     1              [illegible]
      [illegible]
      [illegible]
      IF ([illegible]) THEN
      VHORIZ=ABS(LINE(2)/[illegible])
      IF (VHORIZ.GT.VHMAX) THEN
      REJ(1) = REJ(1) + 1
      TESTFLAG = 0
      ENDIF
      ENDIF
      IF (LINE(2).EQ.[illegible]) THEN
      REJ(2) = REJ(2) + 1
      TESTFLAG = 0
      ENDIF
      IF (ABS(LINE(2)).GT.VRMAX) THEN
      REJ(3)=REJ(3)+1
      TESTFLAG = 0
      ENDIF
C
C     DETERMINE POLARIZATION
C
      ANGLE=LINE([illegible])-LINE(8)
C
      IF (POLAR.EQ."A") GOTO 10201
C
      IF (POLAR.EQ."L") THEN
C
      IF (ABS(ANGLE).LT.45.0) GO TO 10201
      IF (ABS(ANGLE).GT.135.0.AND.ABS(ANGLE).LT.225.0) GO TO 10201
      IF (ABS(ANGLE).GT.315.0.AND.ABS(ANGLE).LT.360.0) GO TO 10201
      GO TO 10200
      END IF
C
      IF (POLAR.EQ."X") THEN
C
      IF (ANGLE.GT.-135.0.AND.ANGLE.LT.-45.0) GO TO 10201
      IF (ANGLE.GT.225.0.AND.ANGLE.LT.315.0) GO TO 10201
      GO TO 10200
      END IF
C
      IF (POLAR.EQ."O") THEN
C
      IF (ANGLE.GT.45.0.AND.ANGLE.LT.135.0) GO TO 10201
      IF (ANGLE.GT.-315.0.AND.ANGLE.LT.-225.0) GO TO 10201
C
10200 TESTFLAG=0
      REJ(4)=REJ(4)+1
      END IF
C
C     CHECK FOR TOO MANY POINTS
C
```

```
10201 NPOINTS(INDEX)=NPOINTS(INDEX)+1
      IF (NPOINTS(INDEX) .EQ. 2500) THEN
      WRITE (*,*) 'THANKS ANYHOW, BUT IVE ALREADY GOT 2500 POINTS.'
      GO TO 20104
      ENDIF
      IF (TESTFLAG .EQ. 1) THEN
      DO 10202 PARAMETER = 1,7
      SPPZ(INDEX,NPOINTS(INDEX),PARAMETER) = LINE(PARAMETER)
10202 CONTINUE
      TRP(INDEX) = TRP(INDEX) + LINE(5)**2 + LINE(7)**2
      ENDIF
20104 NERR=7
      READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .LT. -990.0) GO TO 20133
      GO TO 20202
20204 IH=IH+1
      FITFLAG = 1
      IF (IH.GT.NHITES) GO TO 20206
      Z=ZMIN+1.5+3.0*(FLOAT(IH-1))
      IF (NPOINTS(IH).EQ.0) THEN
      MISS=MISS+1
      GO TO 20250
      END IF
      DO 2 POINT=1,NPOINTS(IH)
      DO 2 PARAMETER=1,7
      SPP(POINT,PARAMETER)=SPPZ(IH,POINT,PARAMETER)
    2 CONTINUE
C
C  FIT THE SCATTERING POINTS IN THIS WINDOW WITH A 3-VECTOR.
C
20205 CALL WFV
      IF (FITFLAG .EQ. 0) THEN
      NBAD=NBAD+1
      WRITE (*,*) 'VERTICAL FAILURE AT ',IH,NPOINTS(IH),Z
      WRITE (16,90002) Z,(SUCKS(KK),KK=1,8)
      GO TO 20204
      ENDIF
      CALL WFH
      IF (FITFLAG .EQ. 0) THEN
      NBAD=NBAD+1
      WRITE (*,*) 'HORIZONTAL FAILURE AT ',IH,NPOINTS(IH),Z
C
C     WRITE FLAG RECORD FOR THIS ALTITUDE  ( U = 999.0 )
C
20250 WRITE (16,90002) Z,(SUCKS(KK),KK=1,8)
      GO TO 20204
C
C     WRITE GOOD VELOCITY
C
      ELSE
      IF (TRP(IH) .LT. 1) THEN
      TRP(IH) = 0
      ELSE
      TRP(IH) = 10*LOG10(TRP(IH))
      ENDIF
      ENDIF
      CALL PHFIT
      RATE = FLOAT(NPOINTS(IH))/NPROFS
      WRITE (*,90001)
     1 Z,U(IH),V(IH),W(IH),TRP(IH),NPOINTS(IH),NPV,NPH,RATE,
     2 SLOPE,INTERCEPT
90001 FORMAT (1X,F4.0,2(1X,F6.1),2(1X,F5.1),3(1X,I4),3(1X,F5.1))
      X1 = FLOAT(NPOINTS(IH))
```

```
      X2 = FLOAT(NPV)
      X3 = FLOAT(NPH)
      WRITE (16,90002)
     1  Z,U(IH),V(IH),W(IH),TRP(IH),X1,RATE,
     2   SLOPE,INTERCEPT
90002 FORMAT (9(E13.4))
      GO TO 20204
C
C     IT'S NOT TIME TO QUIT; OUTPUT REJECTION STATS TO SCREEN, DATA
C     STATS TO WIND.TXT, AND GO SET UP NEXT OUTFILE
C
20206 CLOSE (16)
      IF (NRAD.EQ.NHITES) THEN
      WRITE (15,90004)
90004 FORMAT (1X/5X," BAD DATA THIS INTERVAL",/)
      WRITE (*,90004)
      END IF
      IF (MISS.EQ.NHITES) THEN
      WRITE (15,90005)
90005 FORMAT (1X/5X," NO DATA THIS INTERVAL",
      WRITE (*,90005)
      GO TO 20101
      ELSE
      WRITE (*,*) 'REJECTIONS:'
      WRITE (*,*) '     VMAX     VR=0     VRMAX     POLAR'
      WRITE (*,102) (REJ(IREJ),IREJ=1,4)
  102 FORMAT (3I8,1X,I8)
      GO TO 20101
      END IF
C
C     TOO BAD - ERROR EXIT
C
90909 WRITE (*,*) ' ERROR EXIT AT NERR = ',NERR,' STATUS =',IO
      GO TO 90950
C
90910 IF (LOOP.LT.3) GO TO 669
C
      CALL REORDER
C
C     LOOKS LIKE WE MAY HAVE SOME WINDS!
C
90940 WRITE (*,*) ' SUCCESSFUL RUN'
90950 CLOSE (15)
      CLOSE (16)
      CLOSE (17)
      CLOSE (18)
      PAUSE ' CR TO EXIT'
      STOP
      END
      SUBROUTINE INNAME
C  INNAME CREATES SSP INPUT FILE.
      INTEGER*4 REJ,IH,NPOINTS(50),INTERVAL,NPV,NPV0,TESTFLAG,FITFLAG,
     1          MINH,MINV,NUMRAD
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      CHARACTER*1 FNUM(3)
      INTEGER*4 YEAR,MONTH,DAY,HOUR,MINUTE,SKIP
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,U(50),V(50),W(50),TRP(50),
     1             REJ(4),LINE(10),
     2             WIDTH(50),IWT(2300),RMSDVR(50),
     4             COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5             NUMRAD(17),VRAD(17)
```

```
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1             NSIGMA,TESTFLAG,IH,NPOINTS,INTERVAL,INFILE,OUTFILE,
     2             INPATH,OUTPATH,NPH,NPV,NPV0,
     3             SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,YEAR,MONTH,DAY,HOUR,MINUTE
C
    1 IFILE=IFILE+1
      SKIP=0
C      CALL IBAD (IFILE,SKIP)
      IF (SKIP.EQ.1) GO TO 1
      I100=IFILE/100
      I10=IFILE/10-10*I100
      I1=IFILE-100*I100-10*I10
      FNUM(1)=CHAR(I100+48)
      FNUM(2)=CHAR(I10+48)
      FNUM(3)=CHAR(I1+48)
      WRITE (INFILE,90003) INPATH,FNUM
90003 FORMAT (A27,3A1)
      RETURN
      END
      SUBROUTINE OUTNAME
C  OUTNAME CREATES OUTPUT FILENAMES.
C
      CHARACTER*2 ASCMONTH,ASCDAY,ASCHOUR,ASCMINUTE
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      INTEGER*4 REJ,IH,NPOINTS(50),INTERVAL,NPV,NPV0,TESTFLAG,FITFLAG,
     1          MINH,MINV,NUMRAD,YEAR,MONTH,DAY,HOUR,MINUTE
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,U(50),V(50),W(50),TRP(50),
     1              REJ(4),LINE(10),
     2              WIDTH(50),IWT(2300),RMSDVR(50),
     4              COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5              NUMRAD(17),VRAD(17)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1             NSIGMA,TESTFLAG,IH,NPOINTS,INTERVAL,INFILE,OUTFILE,
     2             INPATH,OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,YEAR,MONTH,DAY,HOUR,MINUTE
C
      IF (MONTH .LT. 10) THEN
      WRITE (ASCMONTH,90001) '0',MONTH
90001 FORMAT (A1,I1)
      ELSE
      WRITE (ASCMONTH,90002) MONTH
90002 FORMAT (I2)
      ENDIF
      IF (DAY .LT. 10) THEN
      WRITE (ASCDAY,90001) '0',DAY
      ELSE
      WRITE (ASCDAY,90002) DAY
      ENDIF
      IF (HOUR .LT. 10) THEN
      WRITE (ASCHOUR,90001) '0',HOUR
      ELSE
      WRITE (ASCHOUR,90002) HOUR
      ENDIF
      IF (MINUTE .LT. 10) THEN
      WRITE (ASCMINUTE,90001) '0',MINUTE
      ELSE
      WRITE (ASCMINUTE,90002) MINUTE
      ENDIF
      WRITE (OUTFILE,90003)
     1 OUTPATH,ASCMONTH,ASCDAY,ASCHOUR,ASCMINUTE,'.MAW'
```

```
90003 FORMAT (A19,4A2,A4)
      RETURN
      END
      SUBROUTINE WFV
***********************************************************************
C
C   THIS SUBROUTINE CALCULATES THE VERTICAL WINDS FROM MARSTAR DATA.
C   AUGUST 17, 1990
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      INTEGER*4 YEAR,MONTH,DAY,HOUR,MINUTE
      DIMENSION A(3,3),WINDV(3)
      REAL*4 SIGMA,SIGMALAST,PI
      INTEGER*4 FLAG,IZA
      INTEGER*4 REJ,IH,POINT,NPOINTS(50),INTERVAL,NPV,NPV0,TESTFLAG,
     1FITFLAG,MINH,MINV,NUMRAD
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,U(50),V(50),W(50),TRP(50),
     1              REJ(4),LINE(10),
     2              WIDTH(50),IWT(2300),RMSDVR(50),
     4              COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5              NUMRAD(17),VRAD(17)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1             NSIGMA,TESTFLAG,IH,NPOINTS,INTERVAL,INFILE,OUTFILE,
     2             INPATH,OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,YEAR,MONTH,DAY,HOUR,MINUTE
C
      DO 10101 IA = 1,3
      WINDV(IA) = 0.0
      DO 10101 IB = 1,3
      A(IA,IB) = 0.0
10101 CONTINUE
      DO 10102 II = 1,17
      NUMRAD(II) = 0
      VRAD(II) = 0
10102 CONTINUE
      NPV = NPOINTS(IH)
      DO 10201 POINT = 1,NPOINTS(IH)
      IWT(POINT) = 1
      SINZAX = SQRT(SIN(SPP(POINT,3)*PI/180)**2
     1       +      SIN(SPP(POINT,4)*PI/180)**2)
      IF ((SINZAX .LT. SIN(THMINV*PI/180)) .OR.
     1    (SINZAX .GT. SIN(THMAXV*PI/180))) THEN
      IWT(POINT) = 0
      NPV = NPV - 1
      IF (NPV .LT. MINV) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
      COSL(POINT) = SIN(SPP(POINT,3)*PI/180)
      COSM(POINT) = SIN(SPP(POINT,4)*PI/180)
      COSN(POINT) = SQRT(1 - COSL(POINT)**2 - COSM(POINT)**2)
10201 CONTINUE
      SIGMALAST = 1E8
20001 FLAG = 0
      DO 10301 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10301
      A(1,1) = A(1,1) + COSL(POINT)**2
      A(1,2) = A(1,2) + COSL(POINT)*COSM(POINT)
      A(1,3) = A(1,3) + COSL(POINT)*COSN(POINT)
      A(2,2) = A(2,2) + COSM(POINT)**2
      A(2,3) = A(2,3) + COSM(POINT)*COSN(POINT)
```

```
      A(3,3) = A(3,3) + COSN(POINT)**2
      WINDV(1) = WINDV(1) + SPP(POINT,2)*COSL(POINT)
      WINDV(2) = WINDV(2) + SPP(POINT,2)*COSM(POINT)
      WINDV(3) = WINDV(3) + SPP(POINT,2)*COSN(POINT)
10301 CONTINUE
      A(2,1) = A(1,2)
      A(3,1) = A(1,3)
      A(3,2) = A(2,3)
      DET = A(1,1)*A(2,2)*A(3,3) + 2*A(1,2)*A(1,3)*A(2,3) -
     1      A(1,1)*A(2,3)**2 - A(2,2)*A(1,3)**2 - A(3,3)*A(1,2)**2
      IF (ABS(DET) .LT. 1.0E-7) THEN
      WRITE (*,*) 'WFV: NO SOLUTION'
      FITFLAG = 0
      GO TO 90909
      ENDIF
      U(IH) = (WINDV(1)*(A(2,2)*A(3,3) - A(2,3)**2) +
     1     WINDV(2)*(A(2,3)*A(1,3) - A(1,2)*A(3,3)) +
     2     WINDV(3)*(A(1,2)*A(2,3) - A(1,3)*A(2,2))) / DET
      V(IH) = (WINDV(1)*(A(2,3)*A(1,3) - A(1,2)*A(3,3)) +
     1      WINDV(2)*(A(1,1)*A(3,3) - A(1,3)**2) +
     2      WINDV(3)*(A(1,3)*A(1,2) - A(1,1)*A(2,3))) / DET
      W(IH) = (WINDV(1)*(A(1,2)*A(2,3) - A(1,3)*A(2,2)) +
     1      WINDV(2)*(A(1,2)*A(1,3) - A(1,1)*A(2,3)) +
     2      WINDV(3)*(A(1,1)*A(2,2) - A(1,2)**2)) / DET
C  CALCULATE THE STANDARD DEVIATION (SIGMA)
      ERRORSUM = 0
      DO 10401 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10401
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1           - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      ERRORSUM = ERRORSUM + DVR(POINT)**2
10401 CONTINUE
      SIGMA = SQRT(ERRORSUM/NPV)
      DO 10501 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10501
      IF (ABS(DVR(POINT)) .GT. NSIGMA*SIGMA) THEN
      IWT(POINT) = 0
      FLAG = 1
      NPV = NPV - 1
      IF (NPV .LT. MINV) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
10501 CONTINUE
      IF (FLAG .EQ. 0) GO TO 20002
      IF (FLAG .EQ. 1) THEN
      IF (SIGMA .GE. 0.999*SIGMALAST) GO TO 20002
      IF (SIGMA .LE. 0.01) GO TO 20002
      SIGMALAST = SIGMA
      GO TO 20001
      ENDIF
C  GOOD VELOCITY.
20002 IF ( (ABS(U(IH)) .GT. VMAX)      .OR.
     1     (ABS(V(IH)) .GT. VMAX)      .OR.
     2     (ABS(W(IH)) .GT. VMAX/10.0) ) THEN
      WRITE (*,*) 'IH, U, V, W = ',IH,U(IH),V(IH),W(IH)
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (FITFLAG .EQ. 1) THEN
      RMSDVR(IH) = 0
      DO 10601 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10601
```

```
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1            - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      RMSDVR(IH) = RMSDVR(IH) + DVR(POINT)**2
      ZA = (180/PI)*ASIN(SQRT(1-COSN(POINT)**2))
      IZA = INT(ZA) + 1
      IF (IZA .GT. 17) THEN
      WRITE (*,*) 'IZA = ',IZA
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (IZA .EQ. 17) IZA = 16
      VRAD(IZA) = VRAD(IZA) + DVR(POINT)**2
      NUMRAD(IZA) = NUMRAD(IZA) + 1
10601 CONTINUE
      IF (NPV .GT. 0) THEN
      RMSDVR(IH) = SQRT(RMSDVR(IH)/NPV)
      DO 10701 IALPHA = 1,16
      IF (NUMRAD(IALPHA) .EQ. 0) GO TO 10701
C     WRITE (*,*) 'ZA,VRAD,NUMRAD= ',IALPHA,VRAD(IALPHA),NUMRAD(IALPHA
      VRAD(IALPHA) = SQRT(VRAD(IALPHA)/NUMRAD(IALPHA))
10701 CONTINUE
      ENDIF
      ENDIF
90909 RETURN
      END
      SUBROUTINE WFH
***********************************************************************
C
C   THIS SUBROUTINE CALCULATES HORIZONTAL WINDS FROM MAPSTAR SPPS.
C   AUGUST 17, 1990
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      INTEGER*4 REJ,IH,POINT,NPOINTS(50),INTERVAL,NPV,NPV0,TESTFLAG,
     1FITFLAG,MINH,MINV,NUMRAD
      INTEGER*4 YEAR,MONTH,DAY,HOUR,MINUTE
      DIMENSION H(3,3),WIND(3)
      REAL*4 SIGMA,SIGMALAST,PI
      INTEGER*4 FLAG,IZA
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,U(50),V(50),W(50),TRP(50),
     1              REJ(4),LINE(10),
     2              WIDTH(50),IWT(2300),RMSDVR(50),
     4              COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5              NUMRAD(17),VRAD(17)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1             NSIGMA,TESTFLAG,IH,NPOINTS,INTERVAL,INFILE,OUTFILE,
     2             INPATH,OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,YEAR,MONTH,DAY,HOUR,MINUTE
C
      DO 10101 IA = 1,3
      WIND(IA) = 0
      DO 10101 IB = 1,3
      H(IA,IB) = 0
10101 CONTINUE
      DO 10102 II = 1,17
      NUMRAD(II) = 0
      VRAD(II) = 0
10102 CONTINUE
      NPH = NPOINTS(IH)
      DO 10201 POINT = 1,NPOINTS(IH)
      IWT(POINT) = 1
      SINZAX = SQRT(SIN(SPP(POINT,3)*PI/180)**2
     1       +      SIN(SPP(POINT,4)*PI/180)**2)
```

```
      IF ((SINDAX .LT. SIN(THMINH*PI/180)) .OR.
     1    (SINDAX .GT. SIN(THMAXH*PI/180))) THEN
      IWT(POINT) = 0
      NPH = NPH - 1
      IF (NPH .LT. MINH) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
      COSL(POINT) = SIN(SPP(POINT,3)*PI/180)
      COSM(POINT) = SIN(SPP(POINT,4)*PI/180)
      COSN(POINT) = SQRT(1 - COSL(POINT)**2 - COSM(POINT)**2)
10201 CONTINUE
      SIGMALAST = 1E8
20001 FLAG = 0
      DO 10301 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10301
      H(1,1) = H(1,1) + COSL(POINT)**2
      H(1,2) = H(1,2) + COSL(POINT)*COSM(POINT)
      H(2,2) = H(2,2) + COSM(POINT)**2
      WIND(1) = WIND(1) + SPP(POINT,2)*COSL(POINT)
     1                  - COSL(POINT)*W(IH)
      WIND(2) = WIND(2) + SPP(POINT,2)*COSM(POINT)
     1                  - COSM(POINT)*W(IH)
10301 CONTINUE
      H(2,1) = H(1,2)
      DET = H(1,1)*H(2,2) - H(1,2)**2
      IF (ABS(DET) .LT. 1.0E-7) THEN
      WRITE (*,*) 'MVH: NO SOLUTION'
      FITFLAG = 0
      GO TO 90909
      ENDIF
      U(IH) = (WIND(1)*H(2,2) - WIND(2)*H(1,2))/DET
      V(IH) = (H(1,1)*WIND(2) - H(1,2)*WIND(1))/DET
C  CALCULATE THE STANDARD DEVIATION (SIGMA)
      ERRORSUM = 0
      DO 10401 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10401
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1            - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      ERRORSUM = ERRORSUM + DVR(POINT)**2
10401 CONTINUE
      SIGMA = SQRT(ERRORSUM/NPH)
      DO 10501 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10501
      IF (ABS(DVR(POINT)) .GT. NSIGMA*SIGMA) THEN
      IWT(POINT) = 0
      FLAG = 1
      NPH = NPH - 1
      IF (NPH .LT. MINH) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
10501 CONTINUE
      IF (FLAG .EQ. 0) GO TO 20002
      IF (FLAG .EQ. 1) THEN
      IF (SIGMA .GE. 0.999*SIGMALAST) GO TO 20002
      IF (SIGMA .LE. 0.01) GO TO 20002
      SIGMALAST = SIGMA
      GO TO 20001
      ENDIF
C  GOOD VELOCITY.
20002 IF ( (ABS(U(IH)) .GT. VMAX)      .OR.
```

```
     1      (ABS(V(IH)) .GT. VMAX)        .OR.
     2      (ABS(W(IH)) .GT. VMAX/20) ) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (FITFLAG .EQ. 1) THEN
      RMSDVR(IH) = 0
      DO 10601 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10601
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1            - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      RMSDVR(IH) = RMSDVR(IH) + DVR(POINT)**2
      ZA = (180/PI)*ASIN(SQRT(1-COSN(POINT)**2))
      IZA = INT(ZA) + 1
      IF (IZA .GT. 17) THEN
      WRITE (*,*) 'IZA = ',IZA
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (IZA .EQ. 17) IZA = 16
      VRAD(IZA) = VRAD(IZA) + DVR(POINT)**2
      NUMRAD(IZA) = NUMRAD(IZA) + 1
10601 CONTINUE
      IF (NPH .GT. 0) THEN
      RMSDVR(IH) = SQRT(RMSDVR(IH)/NPH)
      DO 10701 IALPHA = 1,16
      IF (NUMRAD(IALPHA) .EQ. 0) GO TO 10701
C     WRITE (*,*) 'ZA,VRAD,NUMRAD= ',IALPHA,VRAD(IALPHA),NUMRAD(IALPHA)
      VRAD(IALPHA) = SQRT(VRAD(IALPHA)/NUMRAD(IALPHA))
10701 CONTINUE
      ENDIF
      ENDIF
90909 RETURN
      END
      SUBROUTINE PHFIT
**********************************************************************
C
C   THIS SUBROUTINE FITS A STRAIGHT LINE TO THE VARIATION OF VELOCITY
C   VARIANCE VS ZENITH ANGLE.
C   JULY 23, 1990
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      COMMON /SPPFILE/ IFILE,YEAR,MONTH,DAY,HOUR,MINUTE
      INTEGER*4 YEAR,MONTH,DAY,HOUR,MINUTE,
     1          REJ,IH,NPOINTS(50),INTERVAL,NPV,NPV0,TESTFLAG,FITFLAG,
     2          MINH,MINV,NUMRAD
      REAL*4 INTERCEPT
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,U(50),V(50),W(50),TRP(50),
     1              REJ(4),LINE(10),
     2              WIDTH(50),IWT(2300),RMSDVR(50),
     4              COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5              NUMRAD(17),VRAD(17)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1             NSIGMA,TESTFLAG,IH,NPOINTS,INTERVAL,INFILE,OUTFILE,
     2             INPATH,OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
C
      SUMVR = 0
      SUMVRPH = 0
      SUMPH = 0
      SUMPH2 = 0
      SUMI = 0
      DO 10101 IALPHA = 1,17
```

```
      IF (NUMRAD(IALPHA) .EQ. 0) GO TO 10101
      ZA = IALPHA - 0.5
      SUMVR = SUMVR + VRAD(IALPHA)
      SUMVRPH = SUMVRPH + VRAD(IALPHA)*ZA
      SUMPH = SUMPH + ZA
      SUMPH2 = SUMPH2 + ZA**2
      SUMI = SUMI + 1
10101 CONTINUE
      IF (SUMI .GT. 0) THEN
      IF (SUMI*SUMPH2 - SUMPH**2 .GT. 0) THEN
      SLOPE = (SUMI*SUMVRPH - SUMVR*SUMPH)/(SUMI*SUMPH2 - SUMPH**2)
      INTERCEPT = (SUMVR - SLOPE*SUMPH)/SUMI
      ELSE
      SLOPE = 0
      INTERCEPT = 0
      ENDIF
      ENDIF
      RETURN
      END
      SUBROUTINE REORDER
C
C     REORDERS OUTPUT FILES IN DESCENDING HEIGHT.
C     THIS PORTION OF THE PROGRAM IS SPECIFIC TO A 43KM HEIGHT RANGE
C
      CHARACTER*1 TAB
      CHARACTER*40 INFILE,OUTFILE
      DIMENSION H(50),U(50),V(50),W(50),TRP(50),
     *XH(50),RT(50),SL(50),ICPT(50)
      INTEGER*4 IFILE,YEAR,MONTH,DAY,HOUR,MINUTE,NPOINTS(50)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1          NSIGMA,TESTFLAG,IH,NPOINTS,INTERVAL,INFILE,OUTFILE
      COMMON /SPPFILE/ IFILE,YEAR,MONTH,DAY,HOUR,MINUTE
C
      TAB=CHAR(9)
      WRITE (*,*) " REORDERING FILES BY DESCENDING HEIGHTS"
      REWIND (17)
90911 READ (17,101,END=90940) YEAR,MONTH,DAY,HOUR,MINUTE
  101 FORMAT (5I2)
      IF (MONTH.EQ.0) GO TO 90940
      CALL OUTNAME
      CLOSE (16)
      NERR=8
      OPEN (16,ERR=90950,FILE=OUTFILE,STATUS="OLD",FORM="FORMATTED")
      IH=1
90912 READ (16,90001,END=90920,) H(IH),U(IH),V(IH),W(IH),TRP(IH),
     *XH(IH),RT(IH),SL(IH),ICPT(IH)
90001 FORMAT (9(E13.4))
      IH=IH+1
      GO TO 90912
90920 REWIND (16)
      J=15
90921 I=J
      K=0
90922 WRITE (16,90002)  H(I),TAB,U(I),TAB,V(I),TAB,W(I),TAB,TRP(I),
     *TAB,XH(I),TAB,RT(I),TAB,SL(I),TAB,ICPT(I)
90002 FORMAT (E13.4,8(A1,E13.4))
      IF (I.EQ.1) GO TO 90911
      K=K+1
      IF (K.EQ.1) THEN
      I=I+28
      GO TO 90922
      END IF
      IF (K.EQ.2) THEN
      I=I-14
```

```
      GO TO 90922
      ELSE
      J=J-1
      GO TO 90921
      END IF
      GO TO 90911
90940 RETURN
90950 WRITE (*,*) " ERROR IN RECORDERING FILES. NERR = ",NERR
      PAUSE " CR TO EXIT"
      STOP
      END
```

## The ISR and IDI Wind Analysis Program ISRIDIIDIG.f

The FORTRAN program ISRIDIIDIG.f is written to comply with the FORTRAN 77 standard. The runtime program has been compiled as a stand alone application using the Absoft FORTRAN II Compiler running in the Apple MPW environment. Execution has been performed on a 16MHz Macintosh IIcx with 8.0Mb of memory running under System 7.1.

The following data folders and files are essential to execution:

ISRDATA folder, containing the ISR data in files of the form SCENE.$.4 where $ is 2 for Scene II and 3 for Scene III data, and file TUREKFILE. TUREKFILE contains a listing of the start and end times and line of sight azimuth of each ISR measurement. Each is associated with a three digit number AAA which is entered during execution, to determine the comparison interval.

IDIGNSEW folder, containing the GROVES analysis program output files XXXXXXTIDE, where XXXXXX are the year, month and day of the 24 hour interval of the comparison. The appropriate file is chosen automatically.

INFILES folder, containing the IDI scattering point parameter data files SPP - GR YYY from which the appropriate IDI wind profile is calculated. YYY is entered during execution.

*OUTFILES folder, where the calculated IDI wind* profile is stored as file ZZZZZZZZ.MAW, where ZZZZZZZZ are interval midpoint month, day, hour and minute.

All these files reside on the disc MAXTOR600. Execution on other platforms will require an appropriate global name change in the source code.

Output is a single tab spaced columns ASCII file AAACRKPLOT, where AAA is the three digit TUREKFILE interval number above.

Defenitions of the column headings follow on the next two pages, and are followed by a sample of the runtime screen.

DEFINITIONS OF GROVES, IDI AND ISR DATA SETS

| | |
|---|---|
| HEIGHT (KM) | |
| IDIG EW | Zonal component from the Groves Analysis, m/s |
| ERR EW | Error, computed from change in Groves over comparison interval and inherent measurement error, m/s |
| IDIG NS | Meridional component |
| ERR NS | Error |
| IDIG W | Vertical component, cm/s |
| ERR W | Error, cm/s |
| IDI EW | Zonal component as measured by the IDI technique, assuming zero vertical velocity |
| ERR EW | Error, computed as change in IDI over comparison interval |
| IDI NS | Meridional component |
| ERR NS | Error |
| IDI W | Vertical component, cm/s |
| ERR W | Error, cm/s |
| ISR EW | Zonal component as measured by the Incoherent Scatter Radar |
| ERR EW | Error, computed from the change in 393° (or 213°) azimuth velocity over the comparison interval, and inherent measurement error |
| ISR NS | Meridional component |
| ERR NS | Error |
| EW IDI-ISR | Modulus of the zonal IDI-ISR velocities |
| NS IDI_ISR | Modulus of the meridional |
| VRG | Groves line of sight in the 393° (or 123°) azimuth direction |
| ERR | Error, scaled from the 3 component error |
| VRP | Groves line of sight in the 303° (or 213°) azimuth direction |
| ERR | Error |
| VRIDI | IDI line of sight in the 393° (or 123°) azimuth direction |
| ERR | Error |
| VPIDI | IDI line of sight in the 303° (or 213°) azimuth direction |
| ERR | Error |

| | |
|---|---|
| VRISR | ISR line of sight in the 393° (or 123°) azimuth direction |
| ERR | Error |
| VPISR | ISR line of sight in the 303° (or 213°) azimuth direction |
| ERR | Error |
| EW IDI | Zonal IDI wind from 3 component calculation |
| NS IDI | Meridional IDI wind from 3 component calculation |

```
PROGRAM ISR1011D1G

ENTER TUREK INTERVAL NUMBER
(THREE FIGURES, WITH LEADING ZERO(ES))

379
 379 CORRESPONDS TO DATE/TIME 8904101200

IF CORRECT ENTER Y, IF INCORRECT, ENTER N

(OR ENTER ANY OTHER KEY TO EXIT)
Y
  PROCESSING ISR DATA


DETERMINE PROFILE TIMING SEQUENCE

ENTER 0, 1 OR 2 FOR EACH PROFILE

  0   READ PROFILE DATA, BUT DO NOT USE

  1   READ AND USE PROFILE DATA

  2   PROFILE MISSING FROM DATA - SKIP

NOTE - ALL SIX MUST BE DEFINED

0 1 2 3 4 5
1 1 1 1 1 1

SEARCHING FILE MAXTOR600:ISR.DATA:SCENE.2.4

     890410      120043          2          5          303
      70.00      -31.35      93.44

  75.0000  29.5000  4.89000  1  1
  76.0000  33.7000  5.56000  1  2
  77.0000  32.8300  6.11000  1  3
  78.0000  31.4200  5.13000  1  4
  79.0000  32.9700  6.25000  1  5
  80.0000  39.3000  6.10000  1  6
  81.0000  44.6600  8.34000  1  7
  82.0000  41.2100  7.73000  1  8
  83.0000  35.1300  9.09000  1  9
  84.0000  33.4700  9.52000  1  10
  85.0000  55.8800  9.84000  1  11
  86.0000  48.9800  10.8600  1  12
  87.0000  57.4200  10.4000  1  13
  88.0000  41.7500  12.2600  1  14
  89.0000  36.1900  10.9300  1  15
  90.0000  1.00000  10.2300  1  16
  91.0000 -18.9800  9.98000  1  17
  92.0000 -43.6200  9.91000  1  18
  93.0000 -45.8100  10.2000  1  19
  94.0000 -49.7300  9.35000  1  20
  95.0000 -44.7600  11.5400  1  21
  96.0000 -62.0700  12.6600  1  22
  97.0000 -52.6500  13.3900  1  23
```

890410 121745 4 5 393
70.00 3.90 36.99

| | | | | |
|---|---|---|---|---|
| 73.0000 | 3.53000 | 2.51000 | 2 | 1 |
| 74.0000 | -14.2200 | 5.59000 | 2 | 2 |
| 75.0000 | -7.90000 | 4.81000 | 2 | 3 |
| 76.0000 | -4.77000 | 4.08000 | 2 | 4 |
| 77.0000 | -15.6600 | 3.83000 | 2 | 5 |
| 78.0000 | -17.0600 | 3.55000 | 2 | 6 |
| 79.0000 | -20.2700 | 3.47000 | 2 | 7 |
| 80.0000 | -18.9800 | 3.25000 | 2 | 8 |
| 81.0000 | -19.3800 | 3.25000 | 2 | 9 |
| 82.0000 | -24.4300 | 4.10000 | 2 | 10 |
| 83.0000 | -32.2300 | 5.64000 | 2 | 11 |
| 84.0000 | -35.0800 | 7.05000 | 2 | 12 |
| 85.0000 | -27.2700 | 7.60000 | 2 | 13 |
| 86.0000 | 5.03000 | 9.25000 | 2 | 14 |
| 87.0000 | 17.9100 | 10.2600 | 2 | 15 |
| 88.0000 | 31.2000 | 8.97000 | 2 | 16 |
| 89.0000 | 28.2700 | 8.00000 | 2 | 17 |
| 90.0000 | 24.2100 | 6.98000 | 2 | 18 |
| 91.0000 | 15.7300 | 6.51000 | 2 | 19 |
| 92.0000 | 5.66000 | 6.75000 | 2 | 20 |
| 93.0000 | 4.48000 | 6.57000 | 2 | 21 |
| 94.0000 | -2.60000 | 7.70000 | 2 | 22 |
| 95.0000 | -1.67000 | 7.89000 | 2 | 23 |
| 96.0000 | -14.3500 | 8.96000 | 2 | 24 |
| 97.0000 | -20.9500 | 11.1500 | 2 | 25 |

890410 124405 4 5 393
70.00 -31.33 25.14

| | | | | |
|---|---|---|---|---|
| 72.0000 | -14.7600 | 3.30000 | 3 | 1 |
| 73.0000 | -15.4400 | 1.35000 | 3 | 2 |
| 75.0000 | -4.97000 | 1.91000 | 3 | 3 |
| 77.0000 | -8.97000 | 12.8500 | 3 | 4 |
| 78.0000 | -9.01000 | 3.49000 | 3 | 5 |
| 79.0000 | -11.2200 | 4.17000 | 3 | 6 |
| 80.0000 | -14.9000 | 6.09000 | 3 | 7 |
| 81.0000 | -27.3400 | 5.19000 | 3 | 8 |
| 82.0000 | -38.9200 | 3.98000 | 3 | 9 |
| 83.0000 | -30.5200 | 3.51000 | 3 | 10 |
| 84.0000 | -32.5700 | 4.36000 | 3 | 11 |
| 85.0000 | -7.69000 | 7.64000 | 3 | 12 |
| 86.0000 | 7.47000 | 7.77000 | 3 | 13 |
| 87.0000 | 27.7600 | 6.12000 | 3 | 14 |
| 88.0000 | 27.0300 | 7.86000 | 3 | 15 |
| 89.0000 | 20.3000 | 8.72000 | 3 | 16 |
| 90.0000 | 5.90000 | 8.43000 | 3 | 17 |
| 91.0000 | 9.68000 | 7.91000 | 3 | 18 |
| 92.0000 | 7.89000 | 6.46000 | 3 | 19 |
| 93.0000 | 3.31000 | 6.96000 | 3 | 20 |
| 94.0000 | -11.3500 | 7.49000 | 3 | 21 |
| 95.0000 | -30.7900 | 7.82000 | 3 | 22 |
| 96.0000 | -47.6100 | 7.76000 | 3 | 23 |
| 97.0000 | -55.1100 | 9.91000 | 3 | 24 |

| | | | | |
|---|---|---|---|---|
| 890410 | 131012 | 4 | 5 | 393 |
| 70.00 | -3.50 | 35.88 | | |

| | | | | |
|---|---|---|---|---|
| 77.0000 | -22.6200 | 14.8600 | 4 | 1 |
| 78.0000 | -23.8100 | 2.57000 | 4 | 2 |
| 80.0000 | -34.7500 | 2.82000 | 4 | 3 |
| 81.0000 | -36.1500 | 4.36000 | 4 | 4 |
| 82.0000 | -37.5700 | 3.34000 | 4 | 5 |
| 83.0000 | -30.0000 | 5.24000 | 4 | 6 |
| 84.0000 | -14.7000 | 7.17000 | 4 | 7 |
| 85.0000 | -.560000 | 8.93000 | 4 | 8 |
| 86.0000 | 11.6000 | 9.89000 | 4 | 9 |
| 87.0000 | 18.3200 | 7.91000 | 4 | 10 |
| 88.0000 | 34.0800 | 7.98000 | 4 | 11 |
| 89.0000 | 36.6900 | 7.28000 | 4 | 12 |
| 90.0000 | 32.1800 | 7.21000 | 4 | 13 |
| 91.0000 | 15.6000 | 6.97000 | 4 | 14 |
| 92.0000 | -9.000000E-02 | 7.76000 | 4 | 15 |
| 93.0000 | -17.7100 | 8.37000 | 4 | 16 |
| 94.0000 | -31.9200 | 8.78000 | 4 | 17 |
| 95.0000 | -55.7700 | 8.44000 | 4 | 18 |
| 96.0000 | -77.4000 | 10.2800 | 4 | 19 |
| 97.0000 | -108.040 | 11.9500 | 4 | 20 |

| | | | | |
|---|---|---|---|---|
| 890410 | 133634 | 4 | 5 | 393 |
| 70.00 | -3.43 | 5.67 | | |

| | | | | |
|---|---|---|---|---|
| 72.0000 | -6.35000 | 2.38000 | 5 | 1 |
| 74.0000 | -9.53000 | 4.85000 | 5 | 2 |
| 76.0000 | -6.80000 | 1.66000 | 5 | 3 |
| 77.0000 | -16.9300 | 4.06000 | 5 | 4 |
| 78.0000 | -27.6200 | 4.25000 | 5 | 5 |
| 79.0000 | -33.6400 | 3.05000 | 5 | 6 |
| 80.0000 | -32.9700 | 3.25000 | 5 | 7 |
| 81.0000 | -35.4100 | 4.64000 | 5 | 8 |
| 83.0000 | -36.7700 | 2.04000 | 5 | 9 |
| 84.0000 | -29.1900 | 6.00000 | 5 | 10 |
| 85.0000 | -20.2900 | 8.17000 | 5 | 11 |
| 86.0000 | 13.4400 | 9.03000 | 5 | 12 |
| 87.0000 | 25.3500 | 7.82000 | 5 | 13 |
| 88.0000 | 26.2500 | 8.15000 | 5 | 14 |
| 89.0000 | 30.1300 | 9.97000 | 5 | 15 |
| 90.0000 | 31.1700 | 9.20000 | 5 | 16 |
| 91.0000 | 20.0600 | 8.10000 | 5 | 17 |
| 92.0000 | 8.24000 | 7.27000 | 5 | 18 |
| 93.0000 | -3.28000 | 8.87000 | 5 | 19 |
| 94.0000 | -23.2200 | 8.61000 | 5 | 20 |
| 95.0000 | -53.6100 | 9.86000 | 5 | 21 |
| 96.0000 | -63.6800 | 10.2000 | 5 | 22 |
| 97.0000 | -66.1900 | 10.2400 | 5 | 23 |

| | | | | |
|---|---|---|---|---|
| 890410 | 140648 | 2 | 5 | 303 |
| 70.00 | -5.49 | 10.97 | | |

| | | | | |
|---|---|---|---|---|
| 72.0000 | 12.2500 | 8.11000 | 6 | 1 |
| 73.0000 | 14.8700 | 6.40000 | 6 | 2 |
| 74.0000 | 18.1700 | 4.60000 | 6 | 3 |
| 78.0000 | 34.7500 | 7.04000 | 6 | 4 |
| 79.0000 | 39.5900 | 9.02000 | 6 | 5 |
| 80.0000 | 37.7000 | 6.16000 | 6 | 6 |
| 81.0000 | 33.3800 | 4.07000 | 6 | 7 |
| 82.0000 | 36.0800 | 4.49000 | 6 | 8 |
| 83.0000 | 55.7500 | 13.9500 | 6 | 9 |
| 84.0000 | 59.5400 | 11.1200 | 6 | 10 |
| 85.0000 | 58.1100 | 8.76000 | 6 | 11 |
| 86.0000 | 48.3900 | 7.57000 | 6 | 12 |
| 87.0000 | 53.0300 | 9.82000 | 6 | 13 |
| 88.0000 | 36.7200 | 9.47000 | 6 | 14 |
| 89.0000 | 19.7800 | 6.70000 | 6 | 15 |
| 90.0000 | 12.6800 | 6.68000 | 6 | 16 |
| 91.0000 | -4.30000 | 7.44000 | 6 | 17 |
| 92.0000 | -13.9100 | 8.27000 | 6 | 18 |
| 93.0000 | -22.8200 | 11.2500 | 6 | 19 |
| 94.0000 | -22.1900 | 9.23000 | 6 | 20 |
| 95.0000 | 22.9000 | 8.70000 | 6 | 21 |
| 96.0000 | -14.7000 | 12.2700 | 6 | 22 |

INTERVAL ACCEPTED

@ 393° AZIMUTH OUTPUT

| | | |
|---|---|---|
| 96.0 | -14.35 | 26.04 |
| 95.0 | -1.67 | 22.97 |
| 94.0 | -2.60 | 13.46 |
| 93.0 | 4.48 | 10.10 |
| 92.0 | 5.66 | 10.17 |
| 91.0 | 15.73 | 11.49 |
| 90.0 | 24.21 | 17.06 |
| 89.0 | 28.27 | 13.91 |
| 88.0 | 31.20 | 13.42 |
| 87.0 | 17.91 | 16.06 |
| 86.0 | 5.03 | 13.32 |
| 85.0 | -27.27 | 18.11 |
| 84.0 | -35.08 | 9.57 |
| 83.0 | -32.23 | 7.59 |
| 82.0 | -24.43 | 11.61 |
| 81.0 | -19.38 | 8.00 |
| 80.0 | -18.98 | 6.92 |
| 79.0 | -20.27 | 8.45 |
| 78.0 | -17.06 | 7.94 |
| 77.0 | -15.66 | 11.07 |
| 76.0 | -4.77 | 4.08 |
| 75.0 | -7.90 | 6.56 |
| 74.0 | -14.22 | 5.59 |
| 73.0 | 3.53 | 12.02 |

303° AZIMUTH OUTPUT

| | | |
|---|---|---|
| 78.0 | 33.08 | 9.02 |
| 79.0 | 36.28 | 11.93 |
| 80.0 | 38.50 | 8.74 |
| 81.0 | 39.02 | 12.24 |
| 82.0 | 38.65 | 9.65 |
| 83.0 | 45.44 | 22.13 |
| 84.0 | 46.51 | 23.54 |
| 85.0 | 56.99 | 13.27 |
| 86.0 | 48.68 | 13.24 |
| 87.0 | 55.22 | 14.64 |
| 88.0 | 39.24 | 15.89 |
| 89.0 | 27.98 | 17.29 |
| 90.0 | 6.84 | 14.75 |
| 91.0 | -11.64 | 16.21 |
| 92.0 | -28.76 | 24.66 |
| 93.0 | -34.31 | 22.25 |
| 94.0 | -35.96 | 23.49 |
| 95.0 | -33.83 | 21.16 |
| 96.0 | -38.38 | 37.85 |

28 19

NS AND EW COMPONENTS

EARLY NS, EW WIND

| HEIGHT | VEW | ERREW | VNS | ERRNS |
|---|---|---|---|---|
| 75.0 | -29.0 | 4.9 | 9.4 | 4.8 |
| 76.0 | -30.9 | 5.2 | 14.4 | 4.6 |
| 77.0 | -36.1 | 5.5 | 4.7 | 4.6 |
| 78.0 | -35.6 | 4.7 | 2.8 | 4.1 |
| 79.0 | -38.7 | 5.6 | 1.0 | 4.5 |
| 80.0 | -43.3 | 5.4 | 5.5 | 4.3 |
| 81.0 | -48.0 | 7.2 | 8.1 | 5.3 |
| 82.0 | -47.9 | 6.9 | 2.0 | 5.4 |
| 83.0 | -47.0 | 8.2 | -7.9 | 6.8 |
| 84.0 | -47.2 | 8.9 | -11.2 | 7.9 |
| 85.0 | -61.7 | 9.2 | 7.6 | 8.3 |
| 86.0 | -38.3 | 10.4 | 30.9 | 9.8 |
| 87.0 | -38.4 | 10.4 | 46.3 | 10.3 |
| 88.0 | -18.0 | 11.4 | 48.9 | 10.1 |
| 89.0 | -15.0 | 10.1 | 43.4 | 9.0 |
| 90.0 | 12.3 | 9.4 | 20.8 | 8.1 |
| 91.0 | 24.5 | 9.1 | 2.9 | 7.7 |
| 92.0 | 39.7 | 9.1 | -19.0 | 7.8 |
| 93.0 | 40.9 | 9.3 | -21.2 | 7.8 |
| 94.0 | 40.3 | 8.9 | -29.3 | 8.2 |
| 95.0 | 36.6 | 10.6 | -25.8 | 9.1 |

LATE NS, EW WIND

| HEIGHT | VEW | ERREW | VNS | ERRNS |
|---|---|---|---|---|
| 72.0 | -13.7 | 10.6 | 1.3 | 9.1 |
| 74.0 | -20.4 | 10.6 | 1.9 | 9.1 |
| 78.0 | -44.2 | 10.6 | -4.2 | 9.1 |
| 79.0 | -51.5 | 10.6 | -6.7 | 9.1 |
| 80.0 | -49.6 | 10.6 | -7.1 | 9.1 |
| 81.0 | -47.3 | 10.6 | -11.5 | 9.1 |
| 83.0 | -66.8 | 10.6 | -.5 | 9.1 |
| 84.0 | -65.8 | 10.6 | 7.9 | 9.1 |
| 85.0 | -59.8 | 10.6 | 14.6 | 9.1 |
| 86.0 | -33.3 | 10.6 | 37.6 | 9.1 |
| 87.0 | -30.7 | 10.6 | 50.1 | 9.1 |
| 88.0 | -16.5 | 10.6 | 42.0 | 9.1 |
| 89.0 | -.2 | 10.6 | 36.0 | 9.1 |
| 90.0 | 6.3 | 10.6 | 33.0 | 9.1 |
| 91.0 | 14.5 | 10.6 | 14.5 | 9.1 |
| 92.0 | 16.2 | 10.6 | -.7 | 9.1 |
| 93.0 | 17.4 | 10.6 | -15.2 | 9.1 |
| 94.0 | 6.0 | 10.6 | -31.6 | 9.1 |
| 95.0 | -10.0 | 10.6 | -57.4 | 9.1 |

ZONAL AND MERIDIONAL ISR WINDS

| @ HEIGHT | VEW | ERREW | VNS | ERRNS |
|---|---|---|---|---|
| 96.0 | 24.4 | 34.8 | -32.9 | 30.0 |
| 95.0 | 27.5 | 21.7 | -19.8 | 22.4 |
| 94.0 | 28.7 | 21.0 | -21.8 | 17.1 |
| 93.0 | 31.2 | 19.5 | -14.9 | 14.8 |
| 92.0 | 27.2 | 21.4 | -10.9 | 15.9 |
| 91.0 | 18.3 | 15.0 | 6.9 | 13.1 |
| 90.0 | 7.4 | 15.5 | 24.0 | 16.4 |
| 89.0 | -8.1 | 16.4 | 39.0 | 15.0 |
| 88.0 | -15.9 | 15.2 | 47.5 | 14.2 |
| 87.0 | -36.6 | 15.1 | 45.1 | 15.6 |
| 86.0 | -38.1 | 13.3 | 30.7 | 13.3 |
| 85.0 | -62.7 | 14.9 | 8.2 | 16.8 |
| 84.0 | -58.1 | 20.4 | -4.1 | 15.1 |
| 83.0 | -55.7 | 19.0 | -2.3 | 13.6 |
| 82.0 | -45.7 | 10.3 | .6 | 11.1 |
| 81.0 | -43.3 | 11.1 | 5.0 | 9.5 |
| 80.0 | -42.6 | 8.2 | 5.1 | 7.5 |
| 79.0 | -41.5 | 11.0 | 2.8 | 9.6 |
| 78.0 | -37.0 | 8.7 | 3.7 | 8.3 |

ISR PROCESSING COMPLETED

GET GROVES VELOCITIES, ERRORS

ENTER T FOR TIDE, G FOR GROOUT INPUT
T

ACCESSING FILE MAXTOR600:IDIGNSEW:890410TIDE

TIDAL WINDS CALCULATED

DETERMINE IDI WINDS

ENTER FIRST SPP - GR FILE NUMBER
239

***** IDI WIND PROFILE *****

INFILE = MAXTOR600:INFILES:SPP - GR 239

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 10. | 45. | 36. | 600. | -999. | -999. |

OUTFILE = MAXTOR600:OUTFILES:04101214.MAW

INFILE = MAXTOR600:INFILES:SPP - GR 240

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 12. | 19. | 40. | 965. | -999. | -999. |

| ALT | U | V | W | TRP | NTOT | NPV | NPH | RATE | SLOPE | INTERCEPT |
|---|---|---|---|---|---|---|---|---|---|---|
| 69. | 5.2 | -11.4 | 0.0 | 117.6 | 671 | 479 | 124 | 33.5 | .1 | 1.5 |
| 72. | -13.9 | -6.5 | 0.0 | 119.2 | 697 | 470 | 130 | 34.8 | .1 | 1.3 |
| 75. | -24.8 | 5.1 | 0.0 | 122.2 | 644 | 365 | 172 | 32.2 | .1 | 1.4 |
| 78. | -34.3 | 3.4 | .1 | 124.4 | 762 | 373 | 267 | 38.1 | .1 | 1.7 |
| 81. | -46.4 | 13.1 | .2 | 126.6 | 829 | 295 | 352 | 41.5 | .1 | 2.3 |
| 84. | -50.2 | 28.3 | .2 | 130.3 | 1129 | 442 | 622 | 56.5 | .2 | 3.3 |
| 87. | -21.7 | 37.0 | -.5 | 137.1 | 1164 | 380 | 763 | 58.2 | .3 | 4.2 |
| 90. | 28.2 | 14.1 | -1.2 | 140.3 | 798 | 214 | 520 | 39.9 | 0.0 | 7.6 |
| 93. | 43.1 | 6.8 | 0.0 | 137.7 | 505 | 111 | 340 | 25.3 | .2 | 4.7 |
| 96. | 48.9 | 1.1 | -.4 | 154.2 | 399 | 196 | 209 | 20.0 | .2 | 5.3 |
| 99. | 29.3 | -4.2 | -.2 | 167.8 | 668 | 535 | 241 | 33.4 | 0.0 | 6.2 |
| 102. | 23.9 | 3.9 | -.1 | 169.0 | 904 | 708 | 333 | 45.2 | .1 | 5.9 |
| 105. | 32.8 | -19.5 | -.6 | 165.8 | 892 | 640 | 374 | 44.6 | .3 | 4.5 |
| 108. | 39.2 | -40.4 | -.8 | 154.7 | 480 | 281 | 222 | 24.0 | 0.0 | 6.6 |
| 111. | 65.5 | -9.5 | .1 | 127.3 | 174 | 99 | 67 | 8.7 | .6 | 3.0 |

REJECTIONS:

| VMAX | VR=0 | VRMAX | POLAR |
|---|---|---|---|
| 2243 | 0 | 0 | 2357 |

INFILE = MAXTOR600:INFILES:SPP - GR 239

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 10. | 45. | 36. | 600. | -999. | -999. |

OUTFILE = MAXTOR600:OUTFILES:04101214.MAW

INFILE = MAXTOR600:INFILES:SPP - GR 240

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 12. | 19. | 40. | 965. | -999. | -999. |

| ALT | U | V | W | TRP | NTOT | NPV | NPH | RATE | SLOPE | INTERCEPT |
|---|---|---|---|---|---|---|---|---|---|---|
| 70. | 1.1 | -10.1 | -.1 | 118.3 | 689 | 370 | 208 | 34.5 | .1 | 1.8 |
| 73. | -17.5 | -2.7 | 0.0 | 120.0 | 726 | 337 | 232 | 36.3 | .1 | 1.6 |
| 76. | -28.2 | 5.5 | .3 | 122.7 | 660 | 331 | 296 | 33.0 | .1 | 1.6 |
| 79. | -37.6 | 4.8 | .5 | 125.2 | 805 | 372 | 401 | 40.3 | .1 | 1.7 |
| 82. | -53.7 | 20.0 | 1.0 | 127.2 | 917 | 278 | 552 | 45.8 | .1 | 2.6 |
| 85. | -40.7 | 39.6 | -.1 | 132.9 | 1199 | 371 | 806 | 60.0 | .1 | 4.9 |
| 88. | -3.3 | 23.9 | -.6 | 138.9 | 1090 | 328 | 745 | 54.5 | .2 | 6.4 |
| 91. | 41.5 | 13.1 | -1.9 | 139.6 | 718 | 175 | 498 | 35.9 | -.1 | 7.2 |
| 94. | 31.1 | 16.7 | .1 | 137.3 | 404 | 84 | 300 | 20.2 | .6 | 3.6 |
| 97. | 50.1 | 3.0 | -.6 | 154.7 | 342 | 172 | 190 | 17.1 | .2 | 4.9 |
| 100. | 27.6 | -1.3 | -.3 | 167.8 | 705 | 522 | 312 | 35.3 | .2 | 4.7 |
| 103. | 26.3 | -5.6 | -.2 | 169.0 | 944 | 683 | 442 | 47.2 | -.1 | 7.1 |
| 106. | 38.4 | -22.6 | -.9 | 165.8 | 860 | 598 | 398 | 43.0 | .2 | 5.4 |
| 109. | 42.6 | -37.2 | -1.1 | 154.7 | 446 | 277 | 231 | 22.3 | .4 | 4.6 |

REJECTIONS:

| VMAX | VR=0 | VRMAX | POLAR |
|---|---|---|---|
| 2158 | 0 | 0 | 2226 |

```
INFILE = MAXTOR600:INFILES:SPP - GR 239
   -999.     89.      4.     10.     10.      45.     36.    600.   -999.   -999.

OUTFILE = MAXTOR600:OUTFILES:04101214.MAW
  ALT     U       V     W    TRP  NTOT  NPV   NPH  RATE  SLOPE INTERCEPT
INFILE = MAXTOR600:INFILES:SPP - GR 240
   -999.     89.      4.     10.     12.      19.     40.    965.   -999.   -999.
   71.   -3.9   -8.0   0.0 118.7  688  379  273  34.4    .1    1.4
   74.  -21.7    1.8    .3 121.6  619  332  277  31.0   0.0    1.9
   77.  -32.6    4.9    .5 124.2  750  358  393  37.5    .1    1.5
   80.  -43.1    5.5   1.0 125.9  772  281  448  38.6    .1    1.9
   83.  -56.0   24.2    .1 128.8 1110  360  715  55.5    .2    2.8
   86.  -32.5   45.1   -.6 136.3 1144  324  795  57.2    .2    4.5
   89.   18.1   16.1  -1.7 140.4  845  175  575  42.2    .1    6.9
   92.   44.1    7.5    .1 137.9  569   83  408  28.5    .1    5.2
   95.   37.5    8.3   -.7 154.1  431  153  277  21.5    .3    4.9
   98.   40.7   -3.6   -.3 167.8  675  509  283  33.8    .2    5.5




  101.   21.6    -.2   -.3 169.0  891  673  401  44.5    .1    5.4
  104.   28.5  -13.8   -.7 165.8  880  575  467  44.0    .1    5.6
  107.   38.5  -39.0  -1.3 154.7  503  283  272  25.1   0.0    6.4
  110.   45.6  -18.6   -.9 127.4  171   82   97   8.6    .3    6.0
REJECTIONS:
    VMAX    VR=0    VRMAX    POLAR
    1879      0       0      1877
 REORDERING FILES BY DESCENDING HEIGHTS
 SUCCESSFUL RUN

 ***** ERROR CALCULATION - PASS 1 *****

INFILE = MAXTOR600:INFILES:SPP - GR 239
   -999.     89.      4.     10.     10.      45.     36.    600.   -999.   -999.

OUTFILE = MAXTOR600:OUTFILES:04101239.MAW
  ALT     U       V     W    TRP  NTOT  NPV   NPH  RATE  SLOPE INTERCEPT
INFILE = MAXTOR600:INFILES:SPP - GR 240
   -999.     89.      4.     10.     12.      19.     40.    965.   -999.   -999.
   69.    1.0   -8.2   0.0 117.6  696  400  327  27.8    .1    1.7
   72.  -17.4   -1.3    .4 120.2  741  404  354  29.6    .1    1.8
   75.  -29.1    6.3    .7 122.9  742  306  438  29.7    .1    1.5
   78.  -41.4    4.9    .8 125.3  939  320  588  37.6    .1    1.7
   81.  -56.0   14.4    .9 127.2 1003  262  661  40.1    .2    2.2
   84.  -52.7   28.1    .2 130.9 1370  391  941  54.8    .1    4.1
   87.  -27.4   34.4   -.8 136.9 1250  314  921  50.0    .3    4.5
   90.   16.5   17.3  -1.4 139.5  781  177  594  31.2   0.0    6.8
   93.   47.0    3.7   -.2 138.1  405   94  332  16.2    .3    3.7
   96.   30.7    4.7   -.6 162.1  765  519  376  30.6   -.3    8.7
   99.   22.9   -9.3   -.5 172.5 1093  890  401  43.7   -.3    8.2
  102.   14.0  -30.1   -.2 170.8 1229  887  536  49.2    .1    6.0
  105.   26.3  -30.8   -.7 159.0  971  507  619  38.8    .4    5.2
  108.   51.8   -5.3   -.5 133.7  214   49  132   8.6   0.0    6.4
  111.   37.2    9.4    .8 131.5  198   65   93   7.9    .5    3.7
REJECTIONS:
    VMAX    VR=0    VRMAX    POLAR
    2273      0       0      2293
```

```
INFILE = MAXTOR600:INFILES:SPP - GR 239
   -999.      89.       4.      10.      10.      45.      36.     600     -999.    -999.

OUTFILE = MAXTOR600:OUTFILES:04101239.MAW
  ALT     U        V      W    TRP  NTOT  NPV  NPH  RATE  SLOPE INTERCEPT
INFILE = MAXTOR600:INFILES:SPP - GR 240
   -999.      89.       4.      10.      12.      19.      40.     965.    -999.    -999.
  70.    -.4    -7.9   -.2 118.4   707   381  362  28.3     .1    1.9
  73.  -18.7     1.2    .5 120.6   786   395  397  31.4    0.0    2.4
  76.  -30.9     8.1    .7 123.5   807   278  485  32.3     .1    1.6
  79.  -45.6     5.2    .8 126.2   968   289  629  38.7     .1    1.9
  82.  -58.9    18.4   1.2 128.4  1122   236  759  44.9     .3    2.1
  85.  -47.7    33.1    .3 133.1  1423   377 1003  56.9     .1    4.8
  88.  -11.8    27.6   -.9 138.0  1117   309  795  44.7     .1    6.0
  91.   32.6    12.5  -1.4 139.4   646   169  512  25.8    0.0    5.8
  94.   37.8     3.4   -.1 140.3   348    90  280  13.9     .2    5.1
  97.   35.4     8.5   -.7 162.3   721   500  359  28.8    -.5   10.0
 100.   18.7   -10.0   -.4 172.5  1132   886  454  45.3    -.1    6.5
 103.   10.9   -35.5   -.1 170.8  1366   885  679  54.6     .1    6.5
 106.   39.0   -23.6   -.8 158.9   811   494  499  32.4     .3    5.8
 109.   40.6     5.8    .5 132.8   202    39  114   8.1     .2    4.9
REJECTIONS:
    VMAX    VR=0    VRMAX    POLAR
    2207      0       0      2180
INFILE = MAXTOR600:INFILES:SPP - GR 239
   -999.      89.       4.      10.      10.      45.      36.     600.    -999.    -999.

OUTFILE = MAXTOR600:OUTFILES:04101239.MAW
  ALT     U        V      W    TRP  NTOT  NPV  NPH  RATE  SLOPE INTERCEPT
INFILE = MAXTOR600:INFILES:SPP - GR 240




   -999.      89.       4.      10.      12.      19.      40.     965.    -999.    -999.
  71.   -4.0    -6.5    .1 119.9   712   374  387  28.5     .1    1.6
  74.  -21.1     2.8    .6 122.5   715   323  412  28.6    0.0    2.2
  77.  -34.7     8.6    .7 125.0   913   297  588  36.5     .1    1.7
  80.  -50.9     4.1   1.3 126.0   928   262  584  37.1     .1    1.7
  83.  -61.1    23.1    .3 129.9  1342   383  897  53.7     .1    3.7
  86.  -37.5    38.5   -.7 136.3  1280   318  937  51.2     .2    4.8
  89.    4.0    22.4  -1.5 139.6   831   181  613  33.2     .1    6.4
  92.   37.8     3.5   -.2 138.1   484    94  393  19.4     .1    4.1
  95.   33.5     6.2   -.7 162.1   786   512  413  31.4     .1    6.7
  98.   27.7    -5.2   -.5 172.5  1081   878  398  43.2    -.5    9.3
 101.   14.4   -21.3   -.1 170.8  1171   889  507  46.8    0.0    6.5
 104.   16.1   -35.4   -.9 159.0  1013   508  659  40.5     .2    5.9
 107.   57.0     3.2   -.5 133.7   228    60  146   9.1     .1    6.3
 110.   36.5    19.5   1.6 131.6   196    44  104   7.8     .6    2.7
REJECTIONS:
    VMAX    VR=0    VRMAX    POLAR
    1904      0       0      1816
 REORDERING FILES BY DESCENDING HEIGHTS
 SUCCESSFUL RUN
```

***** ERROR CALCULATION - PASS 2 *****

INFILE = MAXTOR600:INFILES:SPP - GR 239

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 10. | 45. | 36. | 600. | -999. | -999. |

OUTFILE = MAXTOR600:OUTFILES:04101149.MAW

INFILE = MAXTOR600:INFILES:SPP - GR 240

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 12. | 19. | 40. | 965. | -999. | -999. |

| ALT | U | V | W | TRP | NTOT | NPV | NPH | RATE | SLOPE | INTERCEPT |
|---|---|---|---|---|---|---|---|---|---|---|
| 69. | 3.2 | -9.5 | -.1 | 122.2 | 808 | 425 | 430 | 32.3 | .1 | 1.6 |
| 72. | -22.0 | -3.8 | .4 | 124.2 | 718 | 298 | 441 | 28.7 | .1 | 1.7 |
| 75. | -33.2 | 2.9 | 1.2 | 124.9 | 763 | 228 | 523 | 30.5 | .1 | 1.5 |
| 78. | -41.7 | 3.2 | .9 | 125.1 | 819 | 262 | 494 | 32.8 | .1 | 1.6 |
| 81. | -48.6 | 18.6 | .8 | 126.4 | 999 | 294 | 645 | 40.0 | .2 | 2.3 |
| 84. | -41.8 | 28.2 | 0.0 | 128.9 | 1237 | 325 | 798 | 49.5 | .1 | 4.1 |
| 87. | -15.6 | 27.6 | -.8 | 135.7 | 1457 | 393 | 923 | 58.3 | .3 | 4.5 |
| 90. | 41.7 | 6.5 | -1.0 | 139.6 | 1042 | 208 | 673 | 41.7 | -.1 | 5.7 |
| 93. | 49.2 | 5.8 | -.6 | 136.3 | 700 | 115 | 471 | 28.0 | .2 | 4.6 |
| 96. | 62.1 | 5.6 | -.1 | 133.5 | 396 | 109 | 262 | 15.8 | .4 | 4.2 |
| 99. | 59.1 | -7.0 | 0.0 | 157.1 | 551 | 306 | 316 | 22.0 | 0.0 | 7.1 |
| 102. | 9.2 | -16.6 | -.1 | 169.1 | 1129 | 821 | 521 | 45.2 | -.2 | 9.4 |
| 105. | 32.8 | -27.3 | -.5 | 169.6 | 1325 | 996 | 484 | 53.0 | 0.0 | 7.0 |
| 108. | 46.1 | -34.4 | -.9 | 159.4 | 1180 | 741 | 582 | 47.2 | .4 | 5.3 |
| 111. | 53.1 | -9.7 | 0.0 | 133.0 | 285 | 69 | 164 | 11.4 | .4 | 3.8 |

REJECTIONS:

| VMAX | VR=0 | VRMAX | POLAR |
|---|---|---|---|
| 2567 | 0 | 0 | 2663 |

INFILE = MAXTOR600:INFILES:SPP - GR 239

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 10. | 45. | 36. | 600. | -999. | -999. |

OUTFILE = MAXTOR600:OUTFILES:04101149.MAW

INFILE = MAXTOR600:INFILES:SPP - GR 240

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| -999. | 89. | 4. | 10. | 12. | 19. | 40. | 965. | -999. | -999. |

| ALT | U | V | W | TRP | NTOT | NPV | NPH | RATE | SLOPE | INTERCEPT |
|---|---|---|---|---|---|---|---|---|---|---|
| 70. | 1.2 | -11.0 | -.2 | 122.6 | 828 | 391 | 471 | 33.1 | .1 | 1.8 |
| 73. | -24.2 | -4.5 | .6 | 124.9 | 762 | 287 | 468 | 30.5 | .1 | 1.6 |
| 76. | -31.8 | 4.0 | 1.2 | 125.0 | 754 | 225 | 511 | 30.2 | .1 | 1.9 |
| 79. | -39.3 | 3.2 | .8 | 125.2 | 879 | 275 | 537 | 35.2 | .1 | 1.6 |
| 82. | -47.8 | 16.0 | .9 | 127.0 | 1051 | 308 | 641 | 42.0 | .2 | 1.9 |
| 85. | -36.7 | 33.4 | .2 | 131.1 | 1360 | 310 | 883 | 54.4 | .2 | 3.8 |
| 88. | 4.9 | 18.7 | -.7 | 137.7 | 1365 | 385 | 852 | 54.6 | .1 | 6.1 |
| 91. | 48.2 | 6.2 | -1.1 | 139.2 | 993 | 194 | 626 | 39.7 | -.1 | 5.5 |
| 94. | 54.7 | 5.5 | -.2 | 134.8 | 581 | 83 | 405 | 23.2 | .3 | 4.4 |
| 97. | 53.1 | .3 | 1.1 | 137.3 | 367 | 47 | 301 | 14.7 | .4 | 4.1 |
| 100. | 30.7 | -9.1 | .3 | 157.2 | 508 | 299 | 269 | 20.3 | -.1 | 8.0 |
| 103. | 25.7 | -9.0 | -.1 | 169.1 | 1101 | 883 | 427 | 44.0 | -.3 | 8.1 |
| 106. | 31.6 | -34.3 | -.5 | 169.6 | 1417 | 1029 | 537 | 56.7 | .2 | 5.2 |
| 109. | 51.8 | -24.5 | -1.0 | 159.4 | 1120 | 777 | 592 | 44.8 | .2 | 7.1 |

REJECTIONS:

| VMAX | VR=0 | VRMAX | POLAR |
|---|---|---|---|
| 2473 | 0 | 0 | 2513 |

```
INFILE = MAXTOR600:INFILES:SPP - GR 239
   -999.     89.      4.      10.      10.      45.      36.     600.    -999.    -999.

OUTFILE = MAXTOR600:OUTFILES:04101149.MAW
  ALT     U        V      W   TRP  NTOT  NPV  NPH  RATE  SLOPE INTERCEPT
INFILE = MAXTOR600:INFILES:SPP - GR 240
   -999.     89.      4.      10.      12.      19.      40.     965.    -999      999.
   71.   -8.1  -11.7    .1 124.0  706  279  440  28.2     .1    1.6
   74.  -28.6   -3.1   1.1 124.8  740  255  475  29.6    0.0    2.2
   77.  -34.4    4.6    .9 124.8  814  264  524  32.6    0.0    2.1
   80   -39.7    3.7    .9 125.6  970  310  605  38.8     .1    1.8
   83.  -49.3   20.3    .6 127.6 1160  332  698  46.4     .2    2.2
   86.  -29.3   38.0   -.7 134.4 1408  380  915  56.3     .1    4.8
   89.   30.4    7.4  -1.0 139.7 1079  200  680  43.2    0.0    6.3
   92.   48.7    2.4   0.0 137.1  780   81  505  31.2    0.0    4.1
   95.   56.9    6.3   1.2 132.4  452   42  320  18.1     .5    3.1
   98.   58.4   -4.9    .1 157.1  592  286  360  23.7     .3    5.5
  101    18.2   -9.7   -.1 169.1 1130  850  496  45.2    -.2    9.2
  104.   21.4  -24.7   -.3 169.6 1292 1052  444  51.7    -.2    7.8
  107.   41.5  -31.6   -.8 159.4 1192  785  588  47.7     .3    5.1
  110.   60.2  -37.2   -.8 133.1  295  125  212  11.8     .9    1.9
REJECTIONS:
    VMAX     VR=0     VRMAX     POLAR
    2224       0        0       2178
 REORDERING FILES BY DESCENDING HEIGHTS
 SUCCESSFUL RUN

 ALL DONE. CR TO EXIT
```

```
C     PROGRAM ISRIDIIDIG
C                                JULY 14, 1993
C     USES SUBROUTINES BASED ON IDIGNSEW, IDIWIND AND ISRNSEW
C     TO PRODUCE FILE "XXXCRKPLOT" TO BE IMPORTED AND
C     GRAPHED BY CRICKETGRAPH (SEE MANUAL FOR DETAILS)
C
C     ACCESSES SUBROUTINES
C      FOR INPUT TIMING
C                          TIMED
C                          INTERVAL
C      FOR ISR PROFILES
C                          NSEWISR
C                          RADIAL
C                          NORMAL
C                          PERP
C                          NSEW1
C                          NSEW2
C      FOR GROVES PROFILES
C                          GNSEWIDI
C                          GPROF1
C                          GPROF2
C                          SUBVERT
C      FOR IDI PROFILES
C                          NSEWIDI
C                          IDI
C                          SUBVERT
C                          INNAME
C                          OUTNAME
C                          WFV
C                          WFH
C                          PHFIT
C                          REORDER
C      AND SETS UP OUTPUT
C                          DEVIANT
C
      CHARACTER*1 ATHERE,C4H(4),DUM(3),DUMIN(11),DUMMY(10),NEG,TAB,
     *SPACE(32),GO,YES,NO,POLAR
      CHARACTER*4 B(43,31),CH4,BLANK
      CHARACTER*10 TUREKTIME
      CHARACTER*11 CRKPLOT
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      CHARACTER*40 INFILE,OUTFILE
      INTEGER*4 YEAR,MONTH,DAY,HOUR,MINUTE,HOWLONG
      REAL*4 NINES
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1            NSIGMA,TESTFLAG,IH,NPOINTS(50),INFILE,OUTFILE,INPATH,
     2            OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /HSKP/ ATHERE,DUM,DUMIN
      COMMON /ARRAYS/ A(43,33),B
      COMMON /SPPFILE/ IFILE,NFILE,POLAR
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
      EQUIVALENCE (IDIDATA,DUMIN),(ISRDATA,DUMIN),
     *(CRKPLOT,DUMIN),(CH4,C4H),(TUREKTIME,DUMMY)
      WRITE (*,*) " PROGRAM ISRIDIIDIG"
      WRITE (*,*) " "
      BLANK="    "
```

```
      NEG="-"
      TAB=CHAR(9)
      YES="Y"
      NO="N"
      ATHERE="@"
      NINES=999.0
      DO 1 I=1,32
      SPACE(I)=TAB
    1 CONTINUE
      DO 2 I=1,43
      DO 2 J=1,33
      A(I,J)=NINES
    2 CONTINUE
*****************************************************************
*   COMMUNICATE WITH USER
*****************************************************************
    3 WRITE (*,*) " ENTER TUREK INTERVAL NUMBER"
      WRITE (*,*) " (THREE FIGURES, WITH LEADING ZERO(ES))"
      WRITE (*,*) " "
      READ (*,"(3A1)") DUM
      INTNUM=100*(ICHAR(DUM(1))-48)+10*(ICHAR(DUM(2))-48)+
     *ICHAR(DUM(3))-48
      DO 4 I=1,3
      DUMIN(I)=DUM(I)
    4 CONTINUE
      CALL TIMED
      WRITE (*,*) INTNUM," CORRESPONDS TO DATE/TIME ",TUREKTIME
      WRITE (*,*) " "
      WRITE (*,*) " IF CORRECT ENTER Y, IF INCORRECT, ENTER N"
      WRITE (*,*) " "
      WRITE (*,*) " (OR ENTER ANY OTHER KEY TO EXIT)"
      READ (*,"(A1)") GO
      IF (GO.EQ.YES) GO TO 5
      IF (GO.EQ.NO) GO TO 3
      IF (GO.NE.YES.OR.GO.NE.NO) THEN
      WRITE (*,*) " CHECK TUREK INTERVAL NUMBER"
      PAUSE
      STOP
      END IF
C
C     SET UP ARRAY SEGMENT A(I,14-31) FROM FILE ISR.DATA.SCENE.3.4
C
    5 CALL NSEWISR
C
C     DETERMINE INTERVAL AS PROCESSED BY NSEWISR
C
      CALL INTERVAL
C
C     SET UP ARRAY SEGMENT A(I,2-7) FROM "XXXXTIDE" (GROVES OUTPUT)
C
      CALL GNSEWIDI
C
C     SET UP ARRAY SEGMENT A(I,8-13) FROM SPP FILE
C
      CALL NSEWIDI
C
C     CALCULATE IDI AND ISR NS AND EW DIFFERENCES
C
```

```
      CALL DEVIANT
C
C     OUTPUT FILE "XXXCRKPLOT"
C
      CRKPLOT="   CRKPLOT "
      DO 7 I=1,3
      DUMIN(I)=DUM(I)
    7 CONTINUE
      OPEN (26,FILE=CRKPLOT,FORM="FORMATTED")
      WRITE (26,101) SPACE
  101 FORMAT ("  HEIGHT (KM)",A1,"IDIG EW",A1,"ERR",A1,"IDIG NS",A1,
     *"ERR",A1,"IDIG W",A1,"ERR",A1,"IDI EW",A1,"ERR",A1,"IDI NS",
     *A1,"ERR",A1,"IDI W",A1,"ERR",A1,"ISR EW",A1,"ERR",A1,"ISR NS",
     *A1,"ERR",A1,"EW IDI-ISR",A1,"NS IDI-ISR",
     *A1,"VRG",A1,"ERR",A1,"VPG",A1,"ERR",A1,"VRIDI",A1,"ERR",A1,"VPIDI"
     *,A1,"ERR",A1,"VRISR",A1,"ERR",A1,"VPISR",A1,"ERR",A1,"EW IDI",A1,
     *"NS IDI")
C
C     SET UP CHARACTER ARRAY B(43,33) FOR PRINTING
C
      DO 45 I=1,43
      DO 45 J=1,33
      IF (A(I,J).EQ.NINES) THEN
      CH4=BLANK
      GO TO 44
      END IF
      SIGN=A(I,J)/ABS(A(I,J))
      IA=ABS(A(I,J))
      IF (IA.EQ.0) A(I,J)=A(I,J)+1.0
      IA100=IA/100
      IA10=IA/10-10*IA100
      IA1=IA-100*IA100-10*IA10
      C4H(1)=BLANK
      C4H(2)=CHAR(IA100+48)
      IF (IA100.EQ.0) C4H(2)=BLANK
      C4H(3)=CHAR(IA10+48)
      C4H(4)=CHAR(IA1+48)
      IF (J.EQ.1) GO TO 42
      IF (C4H(2).NE.BLANK) GO TO 42
      IF (IA10.EQ.0) C4H(3)=BLANK
   42 IF (SIGN.GT.0.0) GO TO 44
      DO 43 K=3,1,-1
      IF (C4H(K).NE.BLANK) GO TO 43
      C4H(K)=NEG
      GO TO 44
   43 CONTINUE
   44 B(I,J)=CH4
   45 CONTINUE
      DO 51 I=1,43
      WRITE (26,501) B(I,1),(TAB,B(I,J),J=2,33)
  501 FORMAT (A4,32(A1,A4))
   51 CONTINUE
      PAUSE " ALL DONE. CR TO EXIT"
      CLOSE (26)
      STOP
      END
      SUBROUTINE TIMED
C
```

```
C     DETERMINES TIMES FOR USE IN CALCULATING INTERVAL
C     FROM TUREK NUMBER XXX BY ACCESSING FILE "TUREKFILE"
C
      CHARACTER*1 BLANK,DUMMY(10),FILETIME(12)
      CHARACTER*10 TUREKTIME,DUMFILE
      INTEGER*4 INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,HOWLONG,NOW,
     *STRT2HOUR,STRT3HOUR,STRT2MIN,STRT3MIN,
     *END2HOUR,END3HOUR,END2MIN,END3MIN,AZ1,AZ2
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,HOWLONG
     *,NOW,INTHALF,STRT2HOUR,STRT3HOUR,STRT2MIN,STRT3MIN,
     *END2HOUR,END3HOUR,END2MIN,END3MIN
      COMMON /ANGLES/ AZ1,AZ2
      EQUIVALENCE (DUMFILE,DUMMY)
      MTIME(M,N)=10*(ICHAR(DUMMY(M))-48)+ICHAR(DUMMY(N))-48
      BLANK=" "
      OPEN (18,FILE="MAXTOR600:ISR.DATA:TUREKFILE",STATUS="OLD",
     *FORM="FORMATTED")
      DO 1 I=1,1000
      READ (18,100,END=6) NUM,FILETIME,AZ1
  100 FORMAT (I5,2X,12A1,12X,I3)
      IF (NUM.EQ.INTNUM) GO TO 2
    1 CONTINUE
    2 DO 3 I=1,6
      DUMMY(I)=FILETIME(I)
    3 CONTINUE
      DUMMY(7)=FILETIME(9)
      DUMMY(8)=FILETIME(10)
      DUMMY(9)=FILETIME(11)
      DUMMY(10)=FILETIME(12)
    4 TUREKTIME=DUMFILE
      DO 5 I=1,10
      IF (DUMMY(I).EQ.BLANK) DUMMY(I )="0"
    5 CONTINUE
C
C     DETERMINE YEAR, MONTH, DAY, HOUR, MINUTE
C
      YEAR=MTIME(1,2)
      MONTH=MTIME(3,4)
      DAY=MTIME(5,6)
      HOUR=MTIME(7,8)
      MINUTE=MTIME(9,10)
C
C     READ HOURS, MINUTES TO BE USED TO FIND LENGTH OF INTERVAL
C
      READ (18,101,END=6) STRT2HOUR,STRT2MIN,END2HOUR,END2MIN
  101 FORMAT (15X,2I2,4X,2I2)
      READ (18,101,END=6) STRT3HOUR,STRT3MIN,END3HOUR,END3MIN
      CLOSE (18)
      RETURN
    6 WRITE (*,*) " DID NOT FIND INTERVAL - CHECK TUREK FILENUMBER"
      PAUSE
      STOP
      END
      SUBROUTINE NSEWISR
C
C                                                  MAY 4, 1993
C     READS "MAXTOR600:ISR.DATA:SCENE.$.4" FILE OF ISR DATA,
C     WITH APPROPRIATE FILE ( $ ) AND DATA INTERVAL BEING
```

```
C     CHOSEN BY THE MONTH AND TUREK INTERVAL NUMBER XXX ENTERED;
C     THE PROGRAM OUTPUTS THE NS AND EW WIND COMPONENTS,
C     TOGETHER WITH LINE OF SIGHT, FOR THE
C     CHOSEN INTERVAL AT HEIGHTS FOR WHICH BOTH QUADRATURE
C     COMPONENTS HAVE INSTRUMENTAL ERRORS WHICH ARE < 15 M/S.
C     THE NS AND EW COMPONENTS ARE CALCULATED AS AVERAGES
C     OF THE APPROPRIATE SOUNDINGS, WITH ERRORS
C     CALCULATED AS ONE SIGMA OF THE DATA USED.
C     IF THERE IS NOT AN AVAILABLE QUADRATURE COMPONENT
C     AT THE START OF THE INTERVAL, BUT THERE IS ONE AT THE END,
C     THEN THE 123' OR 393' TUREK NUMBER STARTING THE INTERVAL
C     SHOULD BE ENTERED WHEN REQUESTED.
C     PROGRAM CAN ALSO PROCESS INDIVIDUAL PROFILES, BUT NO
C     NS/EW COMPONENTS WILL RESULT (OBVIOUSLY!).
C
C     REQUIRES SUBROUTINES
C                         NSEW1
C                         NSEW2
C                         NORMAL
C                         PERP
C                         RADIAL
C
      CHARACTER*1 ALPHA1(3),ALPHA2(3),BLANK,D(3),
     *DUMSTAT(30),DUMIN(30),DUMOUT(30),DUMMY(10),LINE(64),
     *TIMES(10),TAB,YES,NO
      CHARACTER*3 ALPH1,ALPH2,AZ,AZIMUTH
      CHARACTER*10 TUREKTIME,DATTIME
      CHARACTER*30 INFILE,OUTFILE,STATFILE
      INTEGER*4 AZ1,AZ2,YEAR,MONTH,DAY,HOUR,MINUTE
      COMMON /ZPERP/ ZP1(60),VP1(60),ERP1(60),
     *               ZP2(60),VP2(60),ERP2(60)
      COMMON /ZRADIAL/  ZR1(60),VR1(60),ERR1(60),
     *                  ZR2(60),VR2(60),ERR2(60),
     *                  ZR3(60),VR3(60),ERR3(60),
     *                  ZR4(60),VR4(60),ERR4(60)
      COMMON /ZAP/ ZRO(60),VRO(60),ERRO(60),
     *             ZPO(60),VPO(60),ERPO(60)
      COMMON /EXTRAS/ ZERO,ALPH1,ALPH2,NR,NP,NMAX,TAB,NAVGE,NPROFS(6),
     *COS33,COS57,D
      COMMON /ARRAYS/ A(43,33)
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
      COMMON /ANGLES/ AZ1,AZ2
      EQUIVALENCE (ALPHA1,ALPH1),(ALPHA2,ALPH2),(STATFILE,DUMSTAT),
     *(OUTFILE,DUMOUT),(LINE(58),AZ),(TIMES,DATTIME),
     *(INFILE,DUMIN),(TUREKTIME,DUMMY)
      LTIME(M,N)=10*(ICHAR(LINE(M))-48)+ICHAR(LINE(N))-48
      BLANK=CHAR(32)
      ERRNEG=-50.0
      NMAX=0
      ZERO=0.0
      RAD=57.29578
      TAB=CHAR(9)
      YES="Y"
      NO="N"
      DO 11 I=1,10
      DUMOUT(I)=BLANK
      DUMSTAT(I)=BLANK
```

```
   11 CONTINUE
      DO 12 I=1,60
      ZP1(I)=ZERO
      ZP2(I)=ZERO
      ZR1(I)=ZERO
      ZR2(I)=ZERO
      ZR3(I)=ZERO
      ZR4(I)=ZERO
      ZPO(I)=ZERO
      ZRO(I)=ZERO
   12 CONTINUE
C
      WRITE (*,*) "   PROCESSING ISR DATA"
C     SELECT DATA SOURCE FILE
C
   14 WRITE (*,*) " "
      INFILE="MAXTOR600:ISR.DATA:SCENE.3.4 "
      IF (MONTH.EQ.3.OR.MONTH.EQ.4) DUMIN(26)="2"
      WRITE (*,*) " "
      OPEN (17,FILE=INFILE,STATUS="OLD",FORM="FORMATTED")
C
C     LOOK ANGLE AZIMUTHS
C     SPECIFIC TO AIDA'89        TAKES CARE OF ORIENTATION
C
      IF (AZ1.EQ.123.OR.AZ1.EQ.213) THEN
      AZ1=213
      AZ2=123
      END IF
      IF (AZ1.EQ.303) AZ2=393
      IF (AZ1.EQ.393) THEN
      AZ1=303
      AZ2=393
      END IF
      IAZ1=AZ1/100
      JAZ1=AZ1/10-10*IAZ1
      KAZ1=AZ1-100*IAZ1-10*JAZ1
      ALPHA1(1)=CHAR(IAZ1+48)
      ALPHA1(2)=CHAR(JAZ1+48)
      ALPHA1(3)=CHAR(KAZ1+48)
      IAZ2=AZ2/100
      JAZ2=AZ2/10-10*IAZ2
      KAZ2=AZ2-100*IAZ2-10*JAZ2
      ALPHA2(1)=CHAR(IAZ2+48)
      ALPHA2(2)=CHAR(JAZ2+48)
      ALPHA2(3)=CHAR(KAZ2+48)
      WRITE (*,*) " "
      WRITE (*,*) " DETERMINE PROFILE TIMING SEQUENCE"
      WRITE (*,*) " "
      WRITE (*,*) " ENTER 0, 1 OR 2 FOR EACH PROFILE"
      WRITE (*,*) " "
      WRITE (*,*) "   0   READ PROFILE DATA, BUT DO NOT USE"
      WRITE (*,*) " "
      WRITE (*,*) "   1   READ AND USE PROFILE DATA"
      WRITE (*,*) " "
      WRITE (*,*) "   2   PROFILE MISSING FROM DATA - SKIP"
      WRITE (*,*) " "
      WRITE (*,*) " NOTE - ALL SIX MUST BE DEFINED"
      WRITE (*,*) " "
```

```
      WRITE (*,*) "0 1 2 3 4 5"
      READ (*,*) NPROFS
      NAVGE=0
      DO 18 I=2,5
      M=NPROFS(I)
      IF (NPROFS(I).EQ.2) M=0
      NAVGE=NAVGE+M
   18 CONTINUE
      WRITE (*,*) " "
      WRITE (*,*) " SEARCHING FILE ",INFILE
      WRITE (*,*) " "
      L=0
    1 READ (17,100,END=80) LINE
  100 FORMAT (64A1)
      AZIMUTH=AZ
      IF (L.GT.0) GO TO 20
      DO 10 MTIME=1,6
      TIMES(MTIME)=LINE(MTIME+6)
   10 CONTINUE
      DO 120 MTIME=7,10
      TIMES(MTIME)=LINE(MTIME+12)
  120 CONTINUE
      IF (DATTIME.NE.TUREKTIME) GO TO 1
   20 L=L+1
      IF (L.EQ.7) GO TO 7
      IF (NPROFS(1).EQ.0.OR.NPROFS(1).EQ.2) GO TO 9
      IF (NPROFS(6).EQ.0.OR.NPROFS(6).EQ.2) GO TO 9
      IF ((L.EQ.1.AND.AZIMUTH.NE.ALPH1)
     *.OR.(L.EQ.6.AND.AZIMUTH.NE.ALPH1)) THEN
      WRITE (*,*) " "
      WRITE (*,104) L,AZIMUTH,ALPH1
  104 FORMAT (" AZIMUTH ERROR ",I3,2X,2A3)
      CLOSE (17)
      PAUSE " CR TO EXIT"
      STOP
      END IF
    9 MY=LTIME(7,8)
      IF (MY.NE.89) THEN
      WRITE (*,*) " ERROR EXIT - MY =",MY," CHECK STATS"
      PAUSE " CR TO EXIT"
      STOP
      END IF
      WRITE (*,100) LINE
      READ (17,100) LINE
      WRITE (*,100) LINE
      READ (17,100) LINE
      WRITE (*,*) " "
      I=0
    2 I=I+1
      GO TO (21,22,23,24,25,26),L
   21 IF (NPROFS(1).EQ.2) THEN
      L=2
      GO TO 22
      END IF
      READ (17,*,END=80) ZP1(I),VP1(I),ERP1(I)
      Z=ZP1(I)
      ERR=ERP1(I)
      GO TO 3
```

```
22 IF (NPROFS(2).EQ.2) THEN
   L=3
   GO TO 23
   END IF
   READ (17,*,END=80) ZR1(I),VR1(I),ERR1(I)
   Z=ZR1(I)
   ERR=ERR1(I)
   GO TO 3
23 IF (NPROFS(3).EQ.2) THEN
   L=4
   GO TO 24
   END IF
   READ (17,*,END=80) ZR2(I),VR2(I),ERR2(I)
   Z=ZR2(I)
   ERR=ERR2(I)
   GO TO 3
24 IF (NPROFS(4).EQ.2) THEN
   L=5
   GO TO 25
   END IF
   READ (17,*,END=80) ZR3(I),VR3(I),ERR3(I)
   Z=ZR3(I)
   ERR=ERR3(I)
   GO TO 3
25 IF (NPROFS(5).EQ.2) THEN
   L=6
   GO TO 26
   END IF
   READ (17,*,END=80) ZR4(I),VR4(I),ERR4(I)
   Z=ZR4(I)
   ERR=ERR4(I)
   GO TO 3
26 IF (NPROFS(6).EQ.2) GO TO 7
   READ (17,*,END=80) ZP2(I),VP2(I),ERP2(I)
   Z=ZP2(I)
   ERR=ERP2(I)
   GO TO 3
 3 IF (Z.LT.999.0) GO TO 4
   I=I-1
   BACKSPACE (17)
   IF (NMAX.LT.I) NMAX=I
   I=0
   GO TO 1
 4 IF (ERR.LT.ZERO.OR.ERR.GT.15.0) THEN
   I=I-1
   GO TO 2
   END IF
   IF (L.NE.1) GO TO 41
   WRITE (*,*) ZP1(I),VP1(I),ERP1(I),L,I
102 FORMAT (F12.1,A1,F14.2,A1,F15.2,2I5)
   GO TO 2
41 IF (L.NE.2) GO TO 42
   WRITE (*,*) ZR1(I),VR1(I),ERR1(I),L,I
   GO TO 2
42 IF (L.NE.3) GO TO 43
   WRITE (*,*) ZR2(I),VR2(I),ERR2(I),L,I
   GO TO 2
43 IF (L.NE.4) GO TO 44
```

```
      WRITE (*,*) ZR3(I),VR3(I),ERR3(I),L,I
      GO TO 2
   44 IF (L.NE.5) GO TO 45
      WRITE (*,*) ZR4(I),VR4(I),ERR4(I),L,I
      GO TO 2
   45 IF (L.NE.6) GO TO 26
      WRITE (*,*) ZP2(I),VP2(I),ERP2(I),L,I
      GO TO 2
    7 WRITE (*,103)
  103 FORMAT (/"     INTERVAL     ACCEPTED"/)
C
C     CALCULATE AVERAGE WINDS FROM PROFILES
C
      CALL RADIAL
C
      IF (NPROFS(1).EQ.1.AND.NPROFS(6).EQ.2) THEN
      CALL NORMAL (ZP1,VP1,ERP1)
      END IF
C
      IF (NPROFS(1).EQ.2.AND.NPROFS(6).EQ.1) THEN
      CALL NORMAL (ZP2,VP2,ERP2)
      END IF
C
      IF (NPROFS(1).EQ.1.AND.NPROFS(6).EQ.1) THEN
      CALL PERP
      END IF
C
      WRITE (*,*) NR,NP
      IF (NR.LT.2.OR.NP.LT.2) THEN
      WRITE (*,*) " INSUFFICIENT HEIGHTS - NO NS-EW CALCULATED;
     * EXECUTION TERMINATED"
      PAUSE " CR TO EXIT"
      STOP
      END IF
      WRITE (*,*) " NS AND EW COMPONENTS"
C
C     DEFINE PROJECTION ANGLES APPROPRIATE TO INPUT AZIMUTHS
C     FOR USE IN DETERMINING ZONAL AND MERIDIONAL WINDS
C
      COS33=COS(33.0/RAD)
      COS57=COS(57.0/RAD)
C
C     CALCULATE NS, EW PROFILES FROM ZP1,ZR1 AND ZP2,ZR4 ONLY
C     IF AVAILABLE (I.E. "EARLY" AND "LATE" COMPONENTS)
C
      IF (NPROFS(1).EQ.1.AND.NPROFS(6).EQ.1) CALL NSEW1
C
C     CALCULATE ZONAL AND MERIDIONAL COMPONENTS
C
      CALL NSEW2
C
   79 WRITE (*,*) " ISR PROCESSING COMPLETED"
      CLOSE (17)
      RETURN
   80 PAUSE " EOF - CHECK STATS FILE. CR TO EXIT"
      CLOSE (17)
      STOP
   90 PAUSE " ZP1=0. CR TO EXIT"
```

```
      CLOSE (17)
      STOP
      END
      SUBROUTINE RADIAL
      DIMENSION ZALL (30)
      CHARACTER*1 TAB
      CHARACTER*3 ALPH1,ALPH2
      COMMON /ZRADIAL/  ZR1(60),VR1(60),ERR1(60),
     *                  ZR2(60),VR2(60),ERR2(60),
     *                  ZR3(60),VR3(60),ERR3(60),
     *                  ZR4(60),VR4(60),ERR4(60)
      COMMON /ZAP/ ZRO(60),VRO(60),ERRO(60),
     *             ZPO(60),VPO(60),ERPO(60)
      COMMON /EXTRAS/ ZERO,ALPH1,ALPH2,NR,NP,NMAX,TAB,NAVGE,NPROFS(6)
      COMMON /ARRAYS/ A(43,33)
C
      SINZEN=SIN(11.3/57.29578)
      HALF=0.5
      WRITE (*,100) ALPH2
  100 FORMAT (" @ ",A3,"" AZIMUTH OUTPUT")
C
      DO 50 I=1,28
      ZALL(I)=97-I
   50 CONTINUE
      ZALL(29)=ZERO
      NR=0
      I=0
C
C     CHOOSE ONLY DATA BETWEEN 69 AND 96KM
C
      IF (NPROFS(2).NE.1) GO TO 56
   54 I=I+1
      NR=NR+1
      IF (ZALL(I).EQ.ZERO) THEN
      NR=NR-1
      GO TO 66
      END IF
      DO 55 II=1,60
      IF (ZR1(II).EQ.ZERO) GO TO 54
      IF (ZALL(I).NE.ZR1(II)) GO TO 55
      ZRO(NR)=ZALL(I)
      VRO(NR)=VR1(II)
      ERRO(NR)=ERR1(II)
      DO 540 JJ=1,60
      IF (ZR2(JJ).EQ.ZERO) GO TO 54
      IF (ZALL(I).NE.ZR2(JJ)) GO TO 540
      ERRO(NR)=ERRO(NR)+HALF*SQRT((VRO(NR)-VR2(JJ))**2
     *+ERR2(JJ)*ERR2(JJ))
  540 CONTINUE
   55 CONTINUE
      GO TO 54
C
   56 IF (NPROFS(5).NE.1) GO TO 66
      NR=0
      I=0
   64 I=I+1
      NR=NR+1
      IF (ZALL(I).EQ.ZERO) THEN
```

```
      NR=NR-1
      GO TO 66
      END IF
      DO 65 II=1,60
      IF (ZR4(II).EQ.ZERO) GO TO 64
      IF (ZALL(I).NE.ZR4(II)) GO TO 65
      ZRO(NR)=ZALL(I)
      VRO(NR)=VR4(II)
      ERRO(NR)=ERR4(II)
      DO 640 JJ=1,60
      IF (ZR3(JJ).EQ.ZERO) GO TO 64
      IF (ZALL(I).NE.ZR3(JJ)) GO TO 640
      ERRO(NR)=ERRO(NR)+HALF*SQRT((VRO(NR)-VR3(JJ))**2
     *+ERR3(JJ)*ERR3(JJ))
  640 CONTINUE
   65 CONTINUE
      GO TO 64
   66 I=0
      DO 67 IJ=1,NR
      IF (ZRO(IJ).NE.ZERO) THEN
      I=I+1
      ZRO(I)=ZRO(IJ)
      VRO(I)=VRO(IJ)
      ERRO(I)=ERRO(IJ)
      VRLOS=VRO(I)*SINZEN
      ERRLOS=ERRO(I)*SINZEN
      IH=112.1-ZRO(I)
      A(IH,28)=VRLOS
      A(IH,29)=ERRLOS
      WRITE (*,102) ZRO(I),TAB,VRO(I),TAB,ERRO(I)
  102 FORMAT (F12.1,A1,F14.2,A1,F15.2)
      END IF
   67 CONTINUE
      ZRO(I+1)=ZERO
   70 RETURN
      END
      SUBROUTINE NORMAL (ZP,VP,ERP)
      DIMENSION ZP(60),VP(60),ERP(60)
      CHARACTER*1 TAB
      CHARACTER*3 ALPH1,ALPH2
      COMMON /ZAP/ ZRO(60),VRO(60),ERRO(60),
     *             ZPO(60),VPO(60),ERPO(60)
      COMMON /EXTRAS/ ZERO,ALPH1,ALPH2,NR,NP,NMAX,TAB
      COMMON /ARRAYS/ A(43,33)
      SINZEN=SIN(11.3/57.29578)
      WRITE (*,106) ALPH1
  106 FORMAT ("  ",A3,"* AZIMUTH OUTPUT")
      DO 50 IK=1,60
      ZPO(IK)=ZP(IK)
      IF (ZPO(IK).EQ.ZERO.OR.ZPO(IK).GT.999.0) GO TO 51
      VPO(IK)=VP(IK)
      ERPO(IK)=ERP(IK)
      VPLOS=VPO(IK)*SINZEN
      ERPLOS=ERPO(IK)*SINZEN
      IH=112.1-ZPO(IK)
      A(IH,30)=VPLOS
      A(IH,31)=ERPLOS
      WRITE (*,101) ZPO(IK),TAB,VPO(IK),TAB,ERPO(IK)
```

```
  101 FORMAT (F12.1,A1,F14.2,A1,F15.2)
   50 CONTINUE
   51 NP=IK-1
      RETURN
      END
      SUBROUTINE PERP
      CHARACTER*1 TAB
      CHARACTER*3 ALPH1,ALPH2
      COMMON /ZPERP/ ZP1(60),VP1(60),ERP1(60),
     *               ZP2(60),VP2(60),ERP2(60)
      COMMON /ZAP/ ZRO(60),VRO(60),ERRO(60),
     *             ZPO(60),VPO(60),ERPO(60)
      COMMON /EXTRAS/ ZERO,ALPH1,ALPH2,NR,NP,NMAX,TAB,NAVGE,NPROFS(6)
      COMMON /ARRAYS/ A(43,33)
C
      WRITE (*,101) ALPH1
  101 FORMAT ("   ",A3," AZIMUTH OUTPUT")
C
      SINZEN=SIN(11.3/57.29578)
      NP=0
      DO 56 I=1,NMAX
      IF (ZP1(I).GT.1000.0) GO TO 60
      IF (ZP1(I).EQ.ZERO) GO TO 60
      DO 55 J=1,NMAX
      IF (ZP2(J).EQ.ZERO) GO TO 56
      IF (ZP1(I).NE.ZP2(J)) GO TO 55
      NP=NP+1
      ZPO(NP)=ZP2(J)
      VPO(NP)=(VP1(I)+VP2(J))/2.0
      ERPO(NP)=SQRT ((VP1(I)-VPO(NP))**2+(VP2(J)-VPO(NP))**2
     *+ERP1(I)**2+ERP2(J)**2)
      VPLOS=VPO(NP)*SINZEN
      ERPLOS=ERPO(NP)*SINZEN
      IH=112.1-ZPO(NP)
      A(IH,30)=VPLOS
      A(IH,31)=ERPLOS
      WRITE (*,102) ZPO(NP),TAB,VPO(NP),TAB,ERPO(NP)
  102 FORMAT (F12.1,A1,F14.2,A1,F15.2)
   55 CONTINUE
   56 CONTINUE
   60 RETURN
      END
      SUBROUTINE NSEW1
      CHARACTER*1 TAB
      CHARACTER*3 ALPH1,ALPH2
      COMMON /ZPERP/ ZP1(60),VP1(60),ERP1(60),
     *               ZP2(60),VP2(60),ERP2(60)
      COMMON /ZRADIAL/  ZR1(60),VR1(60),ERR1(60),
     *                  ZR2(60),VR2(60),ERR2(60),
     *                  ZR3(60),VR3(60),ERR3(60),
     *                  ZR4(60),VR4(60),ERR4(60)
      COMMON /EXTRAS/ ZERO,ALPH1,ALPH2,NR,NP,NMAX,TAB,NAVGE,NPROFS(6),
     *COS33,COS57
C
      IF (NPROFS(1).EQ.0.OR.NPROFS(1).EQ.2) GO TO 69
      WRITE (*,105)
  105 FORMAT (/" EARLY NS, EW WIND",//)
      WRITE (*,106)
```

```
106 FORMAT (/"    HEIGHT    VEW    ERREW       VNS        ERRNS"/)
    I=0
 64 I=I+1
    IF (I.GT.60) GO TO 69
    IF (ZR1(I).GT.95.0) GO TO 64
    IF(ZR1(I).EQ.ZERO) GO TO 69
    DO 65 J=1,60
    IF (ZP1(J).EQ.ZERO) GO TO 64
    IF(ZR1(I).NE.ZP1(J)) GO TO 65
    IF (ALPH1.EQ."303") THEN
    VEW=VR1(I)*COS57-VP1(J)*COS33
    VNS=VR1(I)*COS33+VP1(J)*COS57
    EREW=SQRT ((ERR1(I)*COS57)**2+(ERP1(J)*COS33)**2)
    ERNS=SQRT ((ERR1(I)*COS33)**2+(ERP1(J)*COS57)**2)
    ELSE
    VEW=VR1(I)*COS33-VP1(J)*COS57
    VNS=-VR1(I)*COS57-VP1(J)*COS33
    EREW=SQRT ((ERR1(I)*COS33)**2+(ERP1(J)*COS57)**2)
    ERNS=SQRT ((ERR1(I)*COS57)**2+(ERP1(J)*COS33)**2)
    END IF
    WRITE (*,107) ZP1(J),TAB,VEW,TAB,EREW,TAB,VNS,TAB,ERNS
107 FORMAT (F8.1,4(A1,F8.1))
    GO TO 66
 65 CONTINUE
 66 GO TO 64
 69 IF (NPROFS(6).EQ.0.OR.NPROFS(6).EQ.2) RETURN
    WRITE (*,108)
108 FORMAT (/" LATE NS, EW WIND",//)
    WRITE (*,106)
    I=0
 74 I=I+1
    IF (I.GT.60) GO TO 79
    IF (ZR4(I).GT.95.0) GO TO 74
    IF(ZR4(I).EQ.ZERO) GO TO 79
    DO 75 J=1,60
    IF (ZP2(J).EQ.ZERO) GO TO 74
    IF(ZR4(I).NE.ZP2(J)) GO TO 75
    IF (ALPH1.EQ."303") THEN
    VEW=VR4(I)*COS57-VP2(J)*COS33
    VNS=VR4(I)*COS33+VP2(J)*COS57
    ELSE
    VEW=VR4(I)*COS33-VP2(J)*COS57
    VNS=-VR4(I)*COS57-VP2(J)*COS33
    EREW=SQRT ((ERR4(I)*COS33)**2+(ERP2(J)*COS57)**2)
    ERNS=SQRT ((ERR4(I)*COS57)**2+(ERP2(J)*COS33)**2)
    END IF
    WRITE (*,107) ZP2(J),TAB,VEW,TAB,EREW,TAB,VNS,TAB,ERNS
    GO TO 76
 75 CONTINUE
 76 GO TO 74
 79 RETURN
    END
    SUBROUTINE NSEW2
    CHARACTER*1 D(3),TAB
    CHARACTER*3 ALPH1,ALPH2
    COMMON /ZAP/ ZRO(60),VRO(60),ERRO(60),
   *             ZPO(60),VPO(60),ERPO(60)
    COMMON /EXTRAS/ ZERO,ALPH1,ALPH2,NR,NP,NMAX,TAB,NAVGE,NPROFS(6),
```

```
     *COS33,COS57,D
      COMMON /ARRAYS/ A(43,33)
C
      WRITE (*,100)
  100 FORMAT (//" ZONAL AND MERIDIONAL ISR WINDS"
     *//" @ HEIGHT   VEW    ERREW     VNS      ERRNS"/)
      SIGN=1.0
      IF (ALPH1.EQ."213") SIGN=-1.0
      I=0
   74 I=I+1
      IF(ZRO(I).EQ.ZERO) GO TO 79
      DO 75 J=1,60
      IF (ZPO(J).EQ.ZERO) GO TO 74
      IF(ZRO(I).NE.ZPO(J)) GO TO 75
      IF (ALPH1.EQ."303") THEN
      VEW=VRO(I)*COS57-VPO(J)*COS33
      VNS=VRO(I)*COS33+VPO(J)*COS57
      EREW=SQRT ((ERRO(I)*COS57)**2+(ERPO(J)*COS33)**2)
      ERNS=SQRT ((ERRO(I)*COS33)**2+(ERPO(J)*COS57)**2)
      ELSE
      VEW=VRO(I)*COS33-VPO(J)*COS57
      VNS=-VRO(I)*COS57-VPO(J)*COS33
      EREW=SQRT ((ERRO(I)*COS33)**2+(ERPO(J)*COS57)**2)
      ERNS=SQRT ((ERRO(I)*COS57)**2+(ERPO(J)*COS33)**2)
      END IF
      WRITE (*,101) ZPO(J),TAB,VEW,TAB,EREW,TAB,VNS,TAB,ERNS
  101 FORMAT (F8.1,4(A1,F8.1))
      II=112.1-ZPO(J)
      A(II,14)=VEW
      A(II,15)=EREW
      A(II,16)=VNS
      A(II,17)=ERNS
      GO TO 74
   75 CONTINUE
      GO TO 74
   79 RETURN
      END
      SUBROUTINE INTERVAL
      CHARACTER*10 TUREKTIME
      CHARACTER*27 LIST
      INTEGER*4 INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,HOWLONG,NOW,
     *STRT2HOUR,STRT3HOUR,STRT2MIN,STRT3MIN,
     *END2HOUR,END3HOUR,END2MIN,END3MIN
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,HOWLONG
     *,NOW,INTHALF,STRT2HOUR,STRT3HOUR,STRT2MIN,STRT3MIN,
     *END2HOUR,END3HOUR,END2MIN,END3MIN
      COMMON /EXTRAS/ LIST,NPROFS(6)
C
      IF (NPROFS(1).EQ.1) THEN
      ENDHOUR=END2HOUR
      ENDMIN=END2MIN
      END IF
      IF (NPROFS(5).EQ.1) THEN
      HOUR=STRT2HOUR
      MINUTE=STRT2MIN
      ENDHOUR=END3HOUR
      ENDMIN=END3MIN
      END IF
```

```
C
      HOWLONG=(60*ENDHOUR+ENDMIN)-(60*HOUR+MINUTE)
C
C     DEFINE MIDPOINT TIME, AND CONVERT TO LOCAL MEAN SOLAR
C     ASSUMES DAYTIME INTERVAL I.E. DOES NOT TEST FOR CHANGE OF DAY
C
      INTHALF=HOWLONG/2
      MINUTE=MINUTE+INTHALF-28
      IF (MINUTE.GT.59) THEN
      HOUR=HOUR+1
      MINUTE=MINUTE-60
      END IF
      IF (MINUTE.LT.0) THEN
      HOUR=HOUR-1
      MINUTE=MINUTE+60
      END IF
      RETURN
      END
      SUBROUTINE GNSEWIDI
C                                    MAY 5, 1993
C     CALCULATES INPUT FILE DESIGNATOR XXXXXX AND
C     READS GROVES OUTPUT FILES "XXXXXXTIDE"oOR "XXXXXXGROOUT"
C     USING INTERVAL MIDPOINT  LOCAL MEAN SOLAR TIME
C     TO PRODUCE ZONAL MERIDIONAL AND LINE OF SIGHT
C     PROFILES, WITH ERRORS, AT 1 KM INTERVALS BETWEEN
C     69 AND 111 KM
C
C     REQUIRES SUBROUTINES
C                             GPROF1
C                             GPROF2
C
      DIMENSION STRTEW(50),STRTNS(50),ENDEW(50),ENDNS(50),
     *STRTVERT(50),ENDVERT(50),WINDEW(50),WINDNS(50),VERT(50),
     *ERREW(50),ERRNS(50),ERRVERT(50)

      CHARACTER *40 TIDE,GROOUT
      CHARACTER*10 TUREKTIME
      CHARACTER*1 DUMMY1(25),DUMMY2(25),WHICH
      INTEGER*4 INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,HOWLONG,NOW,
     *YEAR10,YEAR1,MONTH10,MONTH1,IDAY10,IDAY1
      COMMON /ARRAYS/ A(43,33)
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
      EQUIVALENCE (TIDE,DUMMY1)
      EQUIVALENCE (GROOUT,DUMMY2)
C
      WRITE (*,*) " "
      WRITE (*,*) " GET GROVES VELOCITIES, ERRORS"
      WRITE (*,*) " "
      TWO=2.0
      TWOPI=6.28318
      SIXTY=60.0
      RAD=57.29578
      SINZEN=SIN(11.3/RAD)
      COS57=COS(57.0/RAD)
      COS33=COS(33.0/RAD)
C
      T=HOUR+MINUTE/SIXTY
```

```
      HALFINT=FLOAT(HOWLONG)/(TWO*SIXTY)
C
C     DETERMINE U,V AND W
C
C     GET GROVES DATA  (XXXXGROOUT OR XXXXXXTIDE FILE)
C
      WRITE (*,*) " ENTER T FOR TIDE, G FOR GROOUT INPUT"
      READ (*,*) WHICH
      TIDE="MAXTOR600:IDIGNSEW:      TIDE"
      GROOUT="MAXTOR600:IDIGNSEW:      GROOUT"
C
C     CALCULATE GROVES INPUT FILENUMBER
C
      IDAY=DAY
      IF (HOUR.LT.12) IDAY=IDAY-1
      YEAR10=YEAR/10
      YEAR1=YEAR-10*YEAR10
      MONTH10=MONTH/10
      MONTH1=MONTH-10*MONTH10
      IDAY10=IDAY/10
      IDAY1=IDAY-10*IDAY10
      DUMMY1(20)=CHAR(YEAR10+48)
      DUMMY1(21)=CHAR(YEAR1+48)
      DUMMY1(22)=CHAR(MONTH10+48)
      DUMMY1(23)=CHAR(MONTH1+48)
      DUMMY1(24)=CHAR(IDAY10+48)
      DUMMY1(25)=CHAR(IDAY1+48)
      DO 1 I=20,25
      DUMMY2(I)=DUMMY1(I)
    1 CONTINUE
      IF (WHICH.EQ."T") THEN
      OPEN (15,FILE=TIDE,ACTION="READ",FORM="UNFORMATTED")
      WRITE (*,*) " "
      WRITE (*,*) " ACCESSING FILE ",TIDE
      CALL GPROF1 (T,WINDEW)
      CALL GPROF1 (T,WINDNS)
      CALL GPROF1 (T,VERT)
C
C     CALCULATE "ERRORS" IN U AND V BY DETERMINING GROVES
C     PROFILES FOR SAME LENGTH OF INTERVAL CENTERED ON BEGINNING
C     AND END OF ISR INTERVAL
C
      T=T-HALFINT
      REWIND (15)
      CALL GPROF1 (T,STRTEW)
      CALL GPROF1 (T,STRTNS)
      CALL GPROF1 (T,STRTVERT)
      T=T+TWO*HALFINT
      REWIND (15)
      CALL GPROF1 (T,ENDEW)
      CALL GPROF1 (T,ENDNS)
      CALL GPROF1 (T,ENDVERT)
      ELSE
      OPEN (15,FILE=GROOUT,ACTION="READ",FORM="FORMATTED")
      WRITE (*,*) " "
      WRITE (*,*) " ACCESSING FILE ",GROOUT
      CALL GPROF2 (T,WINDEW,WINDNS,VERT)
C
```

```
C     CALCULATE "ERRORS" IN U AND V BY DETERMINING GROVES
C     PROFILES FOR SAME LENGTH OF INTERVAL CENTERED ON BEGINNING
C     AND END OF ISR INTERVAL
C
      T=T-HALFINT
      REWIND (15)
      CALL GPROF2 (T,STRTEW,STRTNS,STRTVERT)
      T=T+TWO*HALFINT
      REWIND (15)
      CALL GPROF2 (T,ENDEW,ENDNS,ENDVERT)
      END IF
      CLOSE (15)
C
C     SAVE NS AND EW GROVES WIND, WITH ERRORS
C     NOTE THAT THESE ARE NOT EQUIVALENT VELOCITIES WITH W=0
C
C
      DO 2 I=1,43
      A(I,1)=112-I
      A(I,2)=WINDEW(I)
      ERREW(I)=ABS (ENDEW(I)-STRTEW(I))/TWO
      A(I,3)=ERREW(I)
      A(I,4)=WINDNS(I)
      ERRNS(I)=ABS (ENDNS(I)-STRTNS(I))/TWO
      A(I,5)=ERRNS(I)
      A(I,6)=VERT(I)
      ERRVERT(I)=ABS (ENDVERT(I)-STRTVERT(I))/TWO
      A(I,7)=ERRVERT(I)
      A(I,21)=SQRT((ERREW(I)*COS57*SINZEN)**2
     *+(ERRNS(I)*COS33*SINZEN)**2)
      A(I,23)=SQRT((ERREW(I)*COS33*SINZEN)**2
     *+(ERRNS(I)*COS57*SINZEN)**2)
    2 CONTINUE
C
C     CALCULATE GROVES LINE OF SIGHT VELOCITY
C
      CALL SUBVERT (WINDEW,WINDNS,VERT,1)
C
      WRITE (*,*) " "
      WRITE (*,*) " TIDAL WINDS CALCULATED"
      WRITE (*,*) " "
      RETURN
      END
      SUBROUTINE GPROF1 (T,WIND)
C                                                 DECEMBER 16, 1992
C     DETERMINES GROVES WIND FROM FILE "XXXXXXTIDE"
C
      DIMENSION AU(4),PH(4),WIND(50)
      TWOPI=6.28318
C
C     SKIP DOWN TO 111KM
C
      DO 1 I=1,5
      READ (15) UO,(AU(J),PH(J),J=1,2)
    1 CONTINUE
C
C     DETERMINE WIND PROFILE
C
```

```
      DO 2 I=1,46
      READ (15) UO,(AU(J),PH(J),J=1,2)
      IF (PH(1).LT.T) PHASE1=PH(1)-T
      IF (PH(1).GE.T) PHASE1=T-PH(1)
      IF (PH(2).LT.T) PHASE2=PH(2)-T
      IF (PH(2).GE.T) PHASE2=T-PH(2)
      WIND(I)=UO+AU(1)*COS((PHASE1/24.0)*TWOPI)
     *+AU(2)*COS((PHASE2/12.0)*TWOPI)
    2 CONTINUE
      RETURN
      END
      SUBROUTINE GPROF2 (T,VX,VY,VW)
C                                                   MARCH 8, 1993
C     READS GROVES WIND FROM FILE "XXXXXXGROOUT"
C
      CHARACTER*8 DIRNVERT,WHATVERT
      CHARACTER*9 DIRNEW,WHATEW
      CHARACTER*11 DIRNNS,WHATNS
      DIMENSION VX(50),VY(50),VW(50),VXL(25),VYL(25),VWL(25)
      IF (T.LT.12) IT=T+24
      IF (T.GE.12) IT=T-11
      DIRNEW="EAST-WEST"
      DIRNNS="NORTH-SOUTH"
      DIRNVERT="VERTICAL"
C
C     SKIP DOWN TO EAST-WEST 111KM
C
    1 READ (15,*) WHATEW
      IF (WHATEW.NE.DIRNEW) GO TO 1
      DO 2 I=1,7
      READ (15,*) WHATEW
    2 CONTINUE
C
      DO 3 K=1,46
      READ (15,100) IH,(VXL(J),J=1,24)
  100 FORMAT (I5,F8.0,23F5.0)
      VX(K)=VXL(IT)
    3 CONTINUE
C
C     SKIP DOWN TO NORTH - SOUTH 111KM
C
    4 READ (15,*) WHATNS
      IF (WHATNS.NE.DIRNNS) GO TO 4
      DO 5 I=1,7
      READ (15,*) WHATNS
    5 CONTINUE
C
      DO 6 K=1,46
      READ (15,100) IH,(VYL(J),J=1,24)
      VY(K)=VYL(IT)
    6 CONTINUE
C
C
C     SKIP DOWN TO VERTICAL 111KM
C
    7 READ (15,*) WHATVERT
      IF (WHATVERT.NE.DIRNVERT) GO TO 7
      DO 8 I=1,8
```

```
      READ (15,*) WHATVERT
    8 CONTINUE
C
      DO 9 K=1,46
      READ (15,100) IH,(VWL(J),J=1,24)
      VW(K)=VWL(IT)
    9 CONTINUE
C
      RETURN
      END
      SUBROUTINE SUBVERT (U,V,W,NGO)
C                                       JULY 11, 1993
C
C     TRANSFER U,V AND W TO ISR LINES OF SIGHT
C     THEN RECOVER NS, EW COMPONENTS, ASSUMING W=0 (C.F. ISR)
C
      INTEGER*4 AZ1,AZ2
      DIMENSION U(50),V(50),W(50)
      COMMON /ANGLES/ AZ1,AZ2
      COMMON /ARRAYS/ A(43,33)
      RAD=57.29578
      ZEN=11.3/RAD
      SINZEN=SIN(ZEN)
      COS33=COS(33.0/RAD)
      COS57=COS(57.0/RAD)
      RAZ1=FLOAT(AZ1)/RAD
      RAZ2=FLOAT(AZ2)/RAD
      DO 4 I=1,43
      IF (U(I).GT.200.0) GO TO 4
      IF (NGO.EQ.1) W(I)=W(I)/100.0
      VR=U(I)*SIN(RAZ2)*SINZEN+V(I)*COS(RAZ2)*SINZEN+W(I)*COS(ZEN)
      VP=U(I)*SIN(RAZ1)*SINZEN+V(I)*COS(RAZ1)*SINZEN+W(I)*COS(ZEN)
      GO TO (1,2),NGO
    1 A(I,20)=VR
      A(I,22)=VP
      GO TO 3
    2 A(I,24)=VR
      A(I,26)=VP
    3 VRHOR=VR/SINZEN
      VPHOR=VP/SINZEN
      IF (AZ1.EQ.303) THEN
      U(I)=VRHOR*COS57-VPHOR*COS33
      V(I)=VRHOR*COS33+VPHOR*COS57
      ELSE
      U(I)=VRHOR*COS33-VPHOR*COS57
      V(I)=-VRHOR*COS57-VPHOR*COS33
      END IF
      IF (NGO.EQ.1) W(I)=W(I)*100.0
    4 CONTINUE
      RETURN
      END
      SUBROUTINE NSEWIDI
C                              MAY 19, 1993
C
C     CALCULATES NS, EW AND VERTICAL IDI WINDS WITH ERROR ESTIMATES
C
      CHARACTER*1 POLAR
      CHARACTER*10 TUREKTIME
```

```
      INTEGER*4 INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,HOWLONG,NOW
      REAL*4 NINES,HALF
      COMMON /IDIVEL/ U(50),V(50),W(50),U1(50),V1(50),W1(50),
     *U2(50),V2(50),W2(50)
      COMMON /ARRAYS/ A(43,33)
      COMMON /SPPFILE/ IFILE,NFILE,POLAR
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
C
      WRITE (*,*) " "
      WRITE (*,*) " DETERMINE IDI WINDS"
      NINES=999.0
      HALF=0.5
      RAD=57.29578
      SINZEN=SIN(11.3/RAD)
      COS57=COS(57.0/RAD)
      COS33=COS(33.0/RAD)
C
C     MORE PARAMETERS
C
      POLAR="O"
C
C     FOLLOWING CAN BE ACTIVATED FOR POLARIZATION TESTS
C
C     CURRENTLY SET TO ACCEPT "O" ONLY
C
C     WRITE (*,*) " "
C     WRITE (*,*) " SELECT POLARIZATION - ENTER O, X, L OR ALL"
C     READ (*,*) POLAR
      WRITE (*,*) " "
      WRITE (*,*) ' ENTER FIRST SPP - GR  FILE NUMBER'
      READ (*,*) NFILE
C
      WRITE (*,*) " "
      WRITE (*,*) " ***** IDI WIND PROFILE *****"
      WRITE (*,*) " "
      CALL IDI (U,V,W)
C
C     SAVE IDI NS AND EW COMPONENTS
C
      DO 1 I=1,43
      IF (ABS(U(I)).GT.200.0) GO TO 1
      A(I,32)=U(I)
      A(I,33)=V(I)
    1 CONTINUE
C
C     TRANSFER TO ISR LINES OF SIGHT
C     THEN RECOVER IDI NS, EW COMPONENTS, ASSUMING W=0 (C.F. ISR)
C
      CALL SUBVERT (U,V,W,2)
C
C     CALCULATE ERROR AS HALF (VELOCITY FOR INTERVAL AFTER
C     MINUS VELOCITY FOR INTERVAL BEFORE) AT EACH ALTITUDE
C
      INTHALF=HOWLONG/2
      MINUTE=MINUTE+INTHALF
      IF (MINUTE.GT.59) THEN
      HOUR=HOUR+1
```

```
      MINUTE=MINUTE-60
      END IF
      WRITE (*,*) " "
      WRITE (*,*) " ***** ERROR CALCULATION - PASS 1 *****"
      WRITE (*,*) " "
      CALL IDI (U1,V1,W1)
      MINUTE=MINUTE-HOWLONG
      IF (MINUTE.LT.0) THEN
      HOUR=HOUR-1
      MINUTE=MINUTE+60
      END IF
      WRITE (*,*) " "
      WRITE (*,*) " ***** ERROR CALCULATION - PASS 2 *****"
      WRITE (*,*) " "
      CALL IDI (U2,V2,W2)
C
C     STORE IDI NS, EW AND VERTICAL COMPONENTS, WITH ERRORS, IN ARRAY A
C
          WRITE (*,*) " "
      DO 2 I=1,43
      IF (ABS(U(I)).GT.200.0) THEN
      U(I)=NINES
      V(I)=NINES
      W(I)=NINES
      GO TO 2
      END IF
      A(I,8)=U(I)
      ERREW=HALF*ABS(U1(I)-U2(I))
      IF (U1(I).EQ.NINES) ERREW=ABS(U(I)-U2(I))
      IF (U2(I).EQ.NINES) ERREW=ABS(U(I)-U1(I))
      IF (ERREW.LT.5.0) ERREW=5.0
      IF (ERREW.GT.50.0) ERREW=0.0 !NO ERROR CALCULATED - NO ERROR BAR
      A(I,9)=ERREW
      A(I,10)=V(I)
      ERRNS=HALF*ABS(V1(I)-V2(I))
      IF (V1(I).EQ.NINES) ERRNS=ABS(V(I)-V2(I))
      IF (V2(I).EQ.NINES) ERRNS=ABS(V(I)-V1(I))
      IF (ERRNS.LT.5.0) ERRNS=5.0
      IF (ERRNS.GT.50.0) ERRNS=0.0 !NO ERROR CALCULATED - NO ERROR BAR
      A(I,11)=ERRNS
      A(I,25)=SQRT((ERREW*COS57*SINZEN)**2+(ERRNS*COS33*SINZEN)**2)
      A(I,27)=SQRT((ERREW*COS33*SINZEN)**2+(ERRNS*COS57*SINZEN)**2)
      ERR=HALF*100.0*ABS(W1(I)-W2(I))
      IF (W1(I).EQ.NINES) ERRW=ABS(W(I)-W2(I))
      IF (W2(I).EQ.NINES) ERRW=ABS(W(I)-W1(I))
      IF (ERRW.GT.900.0) ERRW=0.0  !NO ERROR CALCULATED - NO ERROR BAR
      A(I,12)=100.0*W(I)
      A(I,13)=ERRW
    2 CONTINUE
      RETURN
      END
      SUBROUTINE IDI (U,V,W)
*****************************************************************
*                                                               *
*      IDI WIND-CALCULATION PROGRAM; MAPSTAR RADAR              *
*      COPYRIGHT 1993, HOLODYNE LIMITED 1986.                   *
*      ALL RIGHTS RESERVED.                                     *
*                                                               *
```

```
************************************************************************
C  MAC VERSION 2.00
C                                    MAY 2, 1993.
C  THIS PROGRAM WILL CALCULATE WIND PROFILES IN 1-KM STEPS, WITH
C  SMOOTHING, FROM REGULAR SCATTERING-POINT PARAMETER HEIGHT
C  PROFILE FILES, TYPE SPP - GR XXX
C
C  CURRENTLY SET UP FOR 69-111KM (15 HEIGHTS); NEED TO CHANGE IH
C  DIMENSION OF SPPZ(IH,I,J), AND SOME CODE, TO MATCH HEIGHT RANGE.
C
C  THE SCATTERING-POINT PARAMETERS ARE :
C  1.  ALTITUDE (KM).
C  2.  RADIAL VELOCITY (M/SEC).
C  3.  ZENITH ANGLE IN EAST-WEST MERIDIAN (DEGREES).
C  4.  ZENITH ANGLE IN NORTH-SOUTH MERIDIAN (DEGREES).
C  5.  VOLTAGE AMPLITUDE ON #1 DIPOLES.
C  6.  PHASE OF #1 DIPOLES (DEGREES).
C  7.  VOLTAGE AMPLITUDE ON #2 DIPOLES.
C  8.  PHASE OF #2 DIPOLES (DEGREES).
C  9.  SPARE
C 10.  SPARE
C EXPLANATION OF EASILY-REPROGRAMMED PARAMETERS (JUST CHANGE THE SOURCE
C  CODE VALUE GIVEN BELOW:
C   VMAX IS THE LARGEST ALLOWED HORIZONTAL VELOCITY. WE TEST EACH POINT
C  AGAINST VMAX BY PROJECTING ITS RADIAL VELOCITY INTO THE HORIZONTAL
C  PLANE, AND REJECT IT IF IT'S BIGGER THAN VMAX.
C    THMAX IS THE LARGEST ACCEPTABLE RADIAL ZENITH ANGLE.
C    THMIN IS THE SMALLEST ACCEPTABLE RADIAL ZENITH ANGLE.
C    MINH, MINV ARE THE MINIMUM NUMBER OF POINTS.  IF THERE ARE NOT
C  SUFFICIENT POINTS, THAT ALTITUDE IS SKIPPED.
C   NSIGMA IS THE MAXIMUM NUMBER OF STANDARD DEVIATIONS FROM THE FIT AN
C  INDIVIDUAL POINT CAN LIE WITHOUT BEING REJECTED FROM THE VELOCITY
C  CALCULATION.
C    ZMIN IS THE BOTTOM ALTITUDE FOR WHICH WINDS ARE TO BE CALCULATED.
C    ZMAX IS THE TOP ALTITUDE FOR WHICH WINDS ARE TO BE CALCULATED.
C WIND CALLS INNAME, OUTNAME, WFV, WFH, PHFIT AND SORT.
      REAL*4 PI,VMAX,THMAXV,U(50),V(50),W(50),TRP(50),SUCKS(8),
     1       LINE(10),RMSDVR(50),COSL(2300),COSM(2300),COSN(2300),
     2       DVR(2300),SLOPE,INTERCEPT,VRAD(17)
      INTEGER*4 REJ(4),IH,PARAMETER,TESTFLAG,POINT,NPROFS,NHITES,EARLY,
     1          NPOINTS(50),HOWLONG,BIGTIME,NPV,NPV0,FITFLAG,MISS,NBAD,
     2          YEAR,MONTH,DAY,HOUR,MINUTE,MINH,MINV,MSEC,IO,NGO,NFILE,
     3          NUMRAD(17),MY,MO,JO,LTIMH,LTIMM,NOW,NOWSTART,NOWEND
      CHARACTER*40 INFILE
      CHARACTER*40 OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      CHARACTER*10 TUREKTIME
      CHARACTER*6 STATE
      CHARACTER*1 POLAR
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,TRP,REJ,LINE,WIDTH(50),IWT(2300),
     2              RMSDVR(50),COSL,COSM,COSN,DVR,NUMRAD,VRAD
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1              NSIGMA,TESTFLAG,IH,NPOINTS,INFILE,OUTFILE,INPATH,
     2              OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,NFILE,POLAR
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
```

```
     *HOWLONG,NOW
C
      PI = 3.14159265
      NTOTAL=0
      VRMAX = 60
      VMAX = 200
      THMAXH = 16
      THMINH = 3
      THMAXV = 5
      THMINV = 0
      THMIN = 0
      MINH = 5
      MINV = 5
      NSIGMA = 3.0
      DO 20000 I=1,8
      SUCKS(I)=999.0
20000 CONTINUE
  668 LOOP=0
C
  669 IFILE=NFILE-1
C
C     PROGRAM ACCEPTS SPP DATA OVER 3KM HEIGHT RANGE FOR EACH ALTITUDE
C     AND LOOPS THREE TIMES THROUGH SPP DATA TO PRODUCE OUTPUT AT 1KM
C     HEIGHT INTERVALS. ZMIN, ZMAX ARE ADJUSTED ACCORDINGLY.
C
      LOOP=LOOP+1
      IF (LOOP.EQ.1) THEN
      STATE="REWIND"
      ZMIN = 67.5
      ZMAX = 112.5
      END IF
      IF (LOOP.EQ.2) THEN
      STATE="APPEND"
      ZMIN = 68.5
      ZMAX = 110.5
      END IF
      IF (LOOP.EQ.3) THEN
      STATE="APPEND"
      ZMIN = 69.5
      ZMAX = 111.5
      END IF
      NHITES=(ZMAX-ZMIN)/3.0+0.1
      NGO=0
      GO TO 203
***********************************************************************
*     RETURN HERE FOR NEW INPUT FILE
***********************************************************************
20203 NGO=1
  203 CALL INNAME
      WRITE (*,*) 'INFILE = ',INFILE
      NERR=1
      OPEN (18,ERR=90909,FILE=INFILE,STATUS='OLD',IOSTAT=IO,
     *FORM='UNFORMATTED')
20100 READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .GT. -990.0) GO TO 20100
      WRITE (*,100) (LINE(KK),KK=1,10)
  100 FORMAT (10F8.0)
```

```
      MY=LINE(2)
      MO=LINE(3)
      JO=LINE(4)
      LTIMH=LINE(5)
      LTIMM=LINE(6)
      MSEC=LINE(7)
      NOWTIME= LTIMM+LTIMH*60+JO*24*60+MO*30*24*60
      REWIND (18)
      IF (NGO.EQ.1) GO TO 20103
************************************************************************
*    SET UP  OUTPUT FILE
************************************************************************
20101 NERR=2
      IF (MONTH.EQ.0) GO TO 90909
      NPROFS=0
      BIGTIME = MINUTE+HOUR*60+DAY*24*60+MONTH*30*24*60
      NOWSTART=BIGTIME-HOWLONG/2
      IF (NGO.EQ.1) GO TO 670
      EARLY=(HOUR+24*(DAY+MONTH*30))-(LTIMH+24*(JO+MO*30))
      IF (EARLY.GT.3) THEN
      WRITE (*,*) " SPP FILE CHOICE EARLY BY ",EARLY," HOURS"
      WRITE (*,*) " NEED TO ENTER LATER SPP FILE NUMBER"
      CLOSE (18)
      READ (*,*) NFILE
      GO TO 668
      END IF
      IF (NOWTIME.GT.NOWSTART) THEN
      WRITE (*,*) " BAD CHOICE OF SPP INPUT FILE; NOWSTART = ",NOWSTART
      WRITE (*,*) "                                  NOWTIME  = ",NOWTIME
      WRITE (*,*) " RE-ENTER SPP INPUT FILE NUMBER"
      CLOSE (18)
      READ (*,*) NFILE
      GO TO 668
      END IF
  670 NOWEND=NOWSTART+HOWLONG
      CALL OUTNAME
      NERR=3
      WRITE (*,*) " "
      WRITE (*,*) 'OUTFILE = ',OUTFILE
      OPEN (16,ERR=90909,FILE=OUTFILE,POSITION=STATE,IOSTAT=IO,
     *FORM="FORMATTED")
10100 DO 10101 IH = 1,NHITES
      NPOINTS(IH)=0
      U(IH) = 0.0
      V(IH) = 0.0
      W(IH) = 0.0
      TRP(IH) = 0.0
      RMSDVR(IH) = 0.0
10101 CONTINUE
      IH=0
      MISS=0
      NBAD=0
      DO 10102 IREJ=1,4
      REJ(IREJ) = 0
10102 CONTINUE
20102 WRITE (*,90003)
90003 FORMAT
     1 (1X,' ALT    U       V      W    TRP  NTOT  NPV  NPH  RATE',
```

```
     2 '  SLOPE INTERCEPT')
20103 NERR=4
      READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .GT. -990.0) GO TO 20103
************************************************************************
*  RETURN TO HERE FOR NEW PROFILE .
************************************************************************
20133 MY=LINE(2)
      MO=LINE(3)
      JO=LINE(4)
      LTIMH=LINE(5)
      LTIMM=LINE(6)
      MSEC=LINE(7)
      NOWTIME= LTIMM+LTIMH*60+JO*24*60+MO*30*24*60
      IF (NOWTIME.LT.NOWSTART) GO TO 20103
      IF (NOWTIME.GT.NOWEND) THEN
      BACKSPACE (18)
      GO TO 20204
      END IF
      NPROFS=NPROFS+1
      NERR=5
      READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .LT. -990.0) THEN
      NPROFS=NPROFS-1
      GO TO 20133
      END IF
C
C     TEST THE POINT FOR:  ALTITUDE
C
20202 IF (LINE(1) .LT. ZMIN) THEN
      NERR=6
      READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
      IF (LINE(1) .LT. -990.0) GO TO 20133
      GO TO 20202
      ENDIF
      IF (LINE(1).GT.ZMAX) GO TO 20103
      INDEX=(LINE(1)-ZMIN)/3.0+1.0
      IF (NPOINTS(INDEX) .GT. 2300) GO TO 20104   !THERE ARE TOO MANY
C
C     TEST FOR:
C                       (1) PROJECTED HORIZONTAL VELOCITY > VMAX,
C                       (2) RADIAL VELOCITY = 0
C                       (3) RADIAL VELOCITY > VRMAX
C                       (3) POLARIZATION
C
C
      TESTFLAG = 1
      COSZAX = SQRT(1.0-(SIN(LINE(3)*PI/180.0))**2
     1            - (SIN(LINE(4)*PI/180.0))**2)
      ZAX=ACOS(COSZAX)
      SINZAX=SIN(ZAX)
      IF (SINZAX .GT. 0.02) THEN
      VHORIZ=ABS(LINE(2)/SINZAX)
      IF(VHORIZ .GT. VMAX) THEN
      REJ(1) = REJ(1) + 1
```

```
      TESTFLAG = 0
      ENDIF
      ENDIF
      IF (LINE(2) .EQ. 0) THEN
      REJ(2) = REJ(2) + 1
      TESTFLAG = 0
      ENDIF
      IF (ABS(LINE(2)) .GT. VRMAX) THEN
      REJ(3)=REJ(3)+1
      TESTFLAG = 0
      ENDIF
C
C     DETERMINE POLARIZATION
C
      ANGLE=LINE(6)-LINE(8)
C
      IF (POLAR.EQ."A") GOTO 10201
C
      IF (POLAR.EQ."L") THEN
C
      IF (ABS(ANGLE).LT.45.0) GO TO 10201
      IF (ABS(ANGLE).GT.135.0.AND.ABS(ANGLE).LT.225.0) GO TO 10201
      IF (ABS(ANGLE).GT.315.0.AND.ABS(ANGLE).LT.360.0) GO TO 10201
      GO TO 10200
      END IF
C
      IF (POLAR.EQ."X") THEN
C
      IF (ANGLE.GT.-135.0.AND.ANGLE.LT.-45.0) GO TO 10201
      IF (ANGLE.GT.225.0.AND.ANGLE.LT.315.0) GO TO 10201
      GO TO 10200
      END IF
C
      IF (POLAR.EQ."O") THEN
C
      IF (ANGLE.GT.45.0.AND.ANGLE.LT.135.0) GO TO 10201
      IF (ANGLE.GT.-315.0.AND.ANGLE.LT.-225.0) GO TO 10201
C
10200 TESTFLAG=0
      REJ(4)=REJ(4)+1
      END IF
C
C     CHECK FOR TOO MANY POINTS
C
10201 NPOINTS(INDEX)=NPOINTS(INDEX)+1
      IF (NPOINTS(INDEX) .EQ. 2300) THEN
      WRITE (*,*) 'THANKS ANYHOW, BUT IVE ALREADY GOT 2300 POINTS.'
      GO TO 20104
      ENDIF
      IF (TESTFLAG .EQ. 1) THEN
      DO 10202 PARAMETER = 1,7
      SPPZ(INDEX,NPOINTS(INDEX),PARAMETER) = LINE(PARAMETER)
10202 CONTINUE
      TRP(INDEX) = TRP(INDEX) + LINE(5)**2 + LINE(7)**2
      ENDIF
20104 NERR=7
      READ (18,ERR=90909,IOSTAT=IO,END=20203) (LINE(PARAMETER),
     *PARAMETER=1,10)
```

```
      IF (LINE(1) .LT. -990.0) GO TO 20133
      GO TO 20202
20204 IH=IH+1
      FITFLAG = 1
      IF (IH.GT.NHITES) GO TO 20206
      Z=ZMIN+1.5+3.0*(FLOAT(IH-1))
      IF (NPOINTS(IH).EQ.0) THEN
      MISS=MISS+1
      GO TO 20250
      END IF
      DO 2 POINT=1,NPOINTS(IH)
      DO 2 PARAMETER=1,7
      SPP(POINT,PARAMETER)=SPPZ(IH,POINT,PARAMETER)
    2 CONTINUE
C
C  FIT THE SCATTERING POINTS IN THIS WINDOW WITH A 3-VECTOR.
C
20205 CALL WFV(U,V,W)
      IF (FITFLAG .EQ. 0) THEN
      NBAD=NBAD+1
      WRITE (*,*) 'VERTICAL FAILURE AT ',IH,NPOINTS(IH),Z
      WRITE (16,90002) Z,(SUCKS(KK),KK=1,8)
      GO TO 20204
      ENDIF
      CALL WFH(U,V,W)
      IF (FITFLAG .EQ. 0) THEN
      NBAD=NBAD+1
      WRITE (*,*) 'HORIZONTAL FAILURE AT ',IH,NPOINTS(IH),Z
C
C     WRITE FLAG RECORD FOR THIS ALTITUDE  ( U = 999.0 )
C
20250 WRITE (16,90002) Z,(SUCKS(KK),KK=1,8)
      GO TO 20204
C
C     WRITE GOOD VELOCITY
C
      ELSE
      IF (TRP(IH) .LT. 1) THEN
      TRP(IH) = 0
      ELSE
      TRP(IH) = 10*LOG10(TRP(IH))
      ENDIF
      ENDIF
      CALL PHFIT
      RATE = FLOAT(NPOINTS(IH))/NPROFS
      WRITE (*,90001)
     1 Z,U(IH),V(IH),W(IH),TRP(IH),NPOINTS(IH),NPV,NPH,RATE,
     2 SLOPE,INTERCEPT
90001 FORMAT (1X,F4.0,2(1X,F6.1),2(1X,F5.1),3(1X,I4),3(1X,F5.1))
      X1 = FLOAT(NPOINTS(IH))
      X2 = FLOAT(NPV)
      X3 = FLOAT(NPH)
      WRITE (16,90002)
     1  Z,U(IH),V(IH),W(IH),TRP(IH),X1,RATE,
     2  SLOPE,INTERCEPT
90002 FORMAT (9(E13.4))
      GO TO 20204
C
```

```
C     IT'S NOT TIME TO QUIT; OUTPUT REJECTION STATS TO SCREEN, DATA
C     STATS TO WIND.TXT, AND GO SET UP NEXT OUTFILE
C
20206 CLOSE (16)
      IF (NBAD.EQ.NHITES) THEN
      WRITE (*,90004)
90004 FORMAT (1X/5X," BAD DATA THIS INTERVAL",/)
      END IF
      IF (MISS.EQ.NHITES) THEN
      WRITE (*,90005)
90005 FORMAT (1X/5X," NO DATA THIS INTERVAL",/)
      GO TO 90910
      ELSE
      WRITE (*,*) 'REJECTIONS:'
      WRITE (*,*) '     VMAX     VR=0     VRMAX     POLAR'
      WRITE (*,102) (REJ(IREJ),IREJ=1,4)
  102 FORMAT (3I8,1X,I8)
      GO TO 90910
      END IF
C
C     TOO BAD - ERROR EXIT
C
90909 WRITE (*,*) ' ERROR EXIT AT NERR = ',NERR,' STATUS =',IO
      GO TO 90950
C
90910 IF (LOOP.LT.3) THEN
      IF (IFILE.EQ.NFILE) CLOSE (18)
      GO TO 669
      END IF
C
      CALL REORDER (U,V,W)
C
C     LOOKS LIKE WE MAY HAVE SOME WINDS!
C
90940 WRITE (*,*) ' SUCCESSFUL RUN'
90950 CLOSE (15)
      CLOSE (16)
      CLOSE (17)
      CLOSE (18)
      RETURN
      END
      SUBROUTINE INNAME
C
C     INNAME CREATES SPP INPUT FILENAME "SPP - GR XXX"
C
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH,OUTPATH
      CHARACTER*1 FNUM(3)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1            NSIGMA,TESTFLAG,IH,NPOINTS(50),INFILE,OUTFILE,INPATH,
     2            OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,NFILE
C
    1 IFILE=IFILE+1
      SKIP=0
C      CALL IBAD (IFILE,SKIP)
      IF (SKIP.EQ.1) GO TO 1
      I100=IFILE/100
```

```
      I10=IFILE/10-10*I100
      I1=IFILE-100*I100-10*I10
      FNUM(1)=CHAR(I100+48)
      FNUM(2)=CHAR(I10+48)
      FNUM(3)=CHAR(I1+48)
      INPATH = 'MAXTOR600:INFILES:SPP - GR '
      WRITE (INFILE,90003) INPATH,FNUM
90003 FORMAT (A27,3A1)
      RETURN
      END
      SUBROUTINE OUTNAME
C
C     OUTNAME CREATES IDI WINDS OUTPUT FILENAME "XXXXXXXX.MAW"
C
      CHARACTER*2 ASCMONTH,ASCDAY,ASCHOUR,ASCMINUTE
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH,OUTPATH
      CHARACTER*10 TUREKTIME
      INTEGER*4 INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,HOWLONG,NOW
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1            NSIGMA,TESTFLAG,IH,NPOINTS(50),INFILE,OUTFILE,INPATH,
     2            OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
C
      IF (MONTH .LT. 10) THEN
      WRITE (ASCMONTH,90001) '0',MONTH
90001 FORMAT (A1,I1)
      ELSE
      WRITE (ASCMONTH,90002) MONTH
90002 FORMAT (I2)
      ENDIF
      IF (DAY .LT. 10) THEN
      WRITE (ASCDAY,90001) '0',DAY
      ELSE
      WRITE (ASCDAY,90002) DAY
      ENDIF
      IF (HOUR .LT. 10) THEN
      WRITE (ASCHOUR,90001) '0',HOUR
      ELSE
      WRITE (ASCHOUR,90002) HOUR
      ENDIF
      IF (MINUTE .LT. 10) THEN
      WRITE (ASCMINUTE,90001) '0',MINUTE
      ELSE
      WRITE (ASCMINUTE,90002) MINUTE
      ENDIF
      OUTPATH = 'MAXTOR600:OUTFILES:'
      WRITE (OUTFILE,90003)
     1 OUTPATH,ASCMONTH,ASCDAY,ASCHOUR,ASCMINUTE,'.MAW'
90003 FORMAT (A19,4A2,A4)
      RETURN
      END
      SUBROUTINE WFV(U,V,W)
*************************************************************************
C
C   THIS SUBROUTINE CALCULATES THE VERTICAL WINDS FROM MAPSTAR SPPS.
C   AUGUST 17, 1990
```

```
C
*************************************************************************
      CHARACTER*1 POLAR
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      CHARACTER*10 TUREKTIME
      DIMENSION A(3,3),WINDV(3)
      REAL*4 SIGMA,SIGMALAST,PI,U(50),V(50),W(50)
      INTEGER*4 FLAG,IZA
      INTEGER*4 REJ,IH,POINT,NPOINTS(50),HOWLONG,NPV,NPV0,TESTFLAG,
     1FITFLAG,MINH,MINV,NUMRAD,YEAR,MONTH,DAY,HOUR,MINUTE,NOW
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,TRP(50),
     1             REJ(4),LINE(10),
     2             WIDTH(50),IWT(2300),RMSDVR(50),
     4             COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5             NUMRAD(17),VRAD(17)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1            NSIGMA,TESTFLAG,IH,NPOINTS,INFILE,OUTFILE,
     2            INPATH,OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,NFILE,POLAR
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
C
      DO 10101 IA = 1,3
      WINDV(IA) = 0.0
      DO 10101 IB = 1,3
      A(IA,IB) = 0.0
10101 CONTINUE
      DO 10102 II = 1,17
      NUMRAD(II) = 0
      VRAD(II) = 0
10102 CONTINUE
      NPV = NPOINTS(IH)
      DO 10201 POINT = 1,NPOINTS(IH)
      IWT(POINT) = 1
      SINZAX = SQRT(SIN(SPP(POINT,3)*PI/180)**2
     1      +      SIN(SPP(POINT,4)*PI/180)**2)
      IF ((SINZAX .LT. SIN(THMINV*PI/180)) .OR.
     1    (SINZAX .GT. SIN(THMAXV*PI/180))) THEN
      IWT(POINT) = 0
      NPV = NPV - 1
      IF (NPV .LT. MINV) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
      COSL(POINT) = SIN(SPP(POINT,3)*PI/180)
      COSM(POINT) = SIN(SPP(POINT,4)*PI/180)
      COSN(POINT) = SQRT(1 - COSL(POINT)**2 - COSM(POINT)**2)
10201 CONTINUE
      SIGMALAST = 1E8
20001 FLAG = 0
      DO 10301 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10301
      A(1,1) = A(1,1) + COSL(POINT)**2
      A(1,2) = A(1,2) + COSL(POINT)*COSM(POINT)
```

```
      A(1,3) = A(1,3) + COSL(POINT)*COSN(POINT)
      A(2,2) = A(2,2) + COSM(POINT)**2
      A(2,3) = A(2,3) + COSM(POINT)*COSN(POINT)
      A(3,3) = A(3,3) + COSN(POINT)**2
      WINDV(1) = WINDV(1) + SPP(POINT,2)*COSL(POINT)
      WINDV(2) = WINDV(2) + SPP(POINT,2)*COSM(POINT)
      WINDV(3) = WINDV(3) + SPP(POINT,2)*COSN(POINT)
10301 CONTINUE
      A(2,1) = A(1,2)
      A(3,1) = A(1,3)
      A(3,2) = A(2,3)
      DET = A(1,1)*A(2,2)*A(3,3) + 2*A(1,2)*A(1,3)*A(2,3) -
     1      A(1,1)*A(2,3)**2 - A(2,2)*A(1,3)**2 - A(3,3)*A(1,2)**2
      IF (ABS(DET) .LT. 1.0E-7) THEN
      WRITE (*,*) 'WFV: NO SOLUTION'
      FITFLAG = 0
      GO TO 90909
      ENDIF
      U(IH) = (WINDV(1)*(A(2,2)*A(3,3)-  A(2,3)**2) +
     1     WINDV(2)*(A(2,3)*A(1,3) - A(1,2)*A(3,3)) +
     2     WINDV(3)*(A(1,2)*A(2,3) - A(1,3)*A(2,2)))/DET
      V(IH) = (WINDV(1)*(A(2,3)*A(1,3) - A(1,2)*A(3,3)) +
     1       WINDV(2)*(A(1,1)*A(3,3) - A(1,3)**2) +
     2       WINDV(3)*(A(1,3)*A(1,2) - A(1,1)*A(2,3)))/DET
      W(IH) = (WINDV(1)*(A(1,2)*A(2,3) - A(1,3)*A(2,2)) +
     1       WINDV(2)*(A(1,2)*A(1,3) - A(1,1)*A(2,3)) +
     2       WINDV(3)*(A(1,1)*A(2,2) - A(1,2)**2))/DET
C  CALCULATE THE STANDARD DEVIATION (SIGMA)
      ERRORSUM = 0
      DO 10401 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10401
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1             - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      ERRORSUM = ERRORSUM + DVR(POINT)**2
10401 CONTINUE
      SIGMA = SQRT(ERRORSUM/NPV)
      DO 10501 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10501
      IF (ABS(DVR(POINT)) .GT. NSIGMA*SIGMA) THEN
      IWT(POINT) = 0
      FLAG = 1
      NPV = NPV - 1
      IF (NPV .LT. MINV) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
10501 CONTINUE
      IF (FLAG .EQ. 0) GO TO 20002
      IF (FLAG .EQ. 1) THEN
      IF (SIGMA .GE. 0.999*SIGMALAST) GO TO 20002
      IF (SIGMA .LE. 0.01) GO TO 20002
      SIGMALAST = SIGMA
      GO TO 20001
      ENDIF
C  GOOD VELOCITY.
20002 IF ( (ABS(U(IH)) .GT. VMAX)      .OR.
     1     (ABS(V(IH)) .GT. VMAX)      .OR.
```

```
     2    (ABS(W(IH)) .GT. VMAX/10.0) ) THEN
      WRITE (*,*) 'IH, U, V, W = ',IH,U(IH),V(IH),W(IH)
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (FITFLAG .EQ. 1) THEN
      RMSDVR(IH) = 0
      DO 10601 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10601
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1            - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      RMSDVR(IH) = RMSDVR(IH) + DVR(POINT)**2
      ZA = (180/PI)*ASIN(SQRT(1-COSN(POINT)**2))
      IZA = INT(ZA) + 1
      IF (IZA .GT. 17) THEN
      WRITE (*,*) 'IZA = ',IZA
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (IZA .EQ. 17) IZA = 16
      VRAD(IZA) = VRAD(IZA) + DVR(POINT)**2
      NUMRAD(IZA) = NUMRAD(IZA) + 1
10601 CONTINUE
      IF (NPV .GT. 0) THEN
      RMSDVR(IH) = SQRT(RMSDVR(IH)/NPV)
      DO 10701 IALPHA = 1,16
      IF (NUMRAD(IALPHA) .EQ. 0) GO TO 10701
C     WRITE (*,*) 'ZA,VRAD,NUMRAD= ',IALPHA,VRAD(IALPHA),NUMRAD(IALPHA)
      VRAD(IALPHA) = SQRT(VRAD(IALPHA)/NUMRAD(IALPHA))
10701 CONTINUE
      ENDIF
      ENDIF
90909 RETURN
      END
      SUBROUTINE WFH(U,V,W)
***********************************************************************
C
C   THIS SUBROUTINE CALCULATES HORIZONTAL WINDS FROM MAPSTAR SPPS.
C   AUGUST 17, 1990
C
***********************************************************************
      CHARACTER*1 POLAR
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      CHARACTER*10 TUREKTIME
      INTEGER*4 REJ,IH,POINT,NPOINTS(50),HOWLONG,NPV,NPV0,TESTFLAG,
     1FITFLAG,MINH,MINV,NUMRAD,YEAR,MONTH,DAY,HOUR,MINUTE,NOW
      DIMENSION H(3,3),WIND(3),U(50),V(50),W(50)
      REAL*4 SIGMA,SIGMALAST,PI
      INTEGER*4 FLAG,IZA
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,TRP(50),
     1             REJ(4),LINE(10),
     2             WIDTH(50),IWT(2300),RMSDVR(50),
     4             COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5             NUMRAD(17),VRAD(17)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
```

```
     1             NSIGMA,TESTFLAG,IH,NPOINTS,INFILE,OUTFILE,
     2             INPATH,OUTPATH,NPH,NPV,NPVO,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,NFILE,POLAR
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
C
      DO 10101 IA = 1 3
      WIND(IA) = 0
      DO 10101 IB = 1,3
      H(IA,IB) = 0
10101 CONTINUE
      DO 10102 II = 1,17
      NUMRAD(II) = 0
      VRAD(II) = 0
10102 CONTINUE
      NPH = NPOINTS(IH)
      DO 10201 POINT = 1,NPOINTS(IH)
      IWT(POINT) = 1
      SINZAX = SQRT(SIN(SPP(POINT,3)*PI/180)**2
     1        +      SIN(SPP(POINT,4)*PI/180)**2)
      IF ((SINZAX .LT. SIN(THMINH*PI/180)) .OR.
     1    (SINZAX .GT. SIN(THMAXH*PI/180))) THEN
      IWT(POINT) = 0
      NPH = NPH - 1
      IF (NPH .LT. MINH) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
      COSL(POINT) = SIN(SPP(POINT,3)*PI/180)
      COSM(POINT) = SIN(SPP(POINT,4)*PI/180)
      COSN(POINT) = SQRT(1 - COSL(POINT)**2 - COSM(POINT)**2)
10201 CONTINUE
      SIGMALAST = 1E8
20001 FLAG = 0
      DO 10301 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10301
      H(1,1) = H(1,1) + COSL(POINT)**2
      H(1,2) = H(1,2) + COSL(POINT)*COSM(POINT)
      H(2,2) = H(2,2) + COSM(POINT)**2
      WIND(1) = WIND(1) + SPP(POINT,2)*COSL(POINT)
     1                  - COSL(POINT)*W(IH)
      WIND(2) = WIND(2) + SPP(POINT,2)*COSM(POINT)
     1                  - COSM(POINT)*W(IH)
10301 CONTINUE
      H(2,1) = H(1,2)
      DET = H(1,1)*H(2,2) - H(1,2)**2
      IF (ABS(DET) .LT. 1.0E-7) THEN
      WRITE (*,*) 'MVH: NO SOLUTION'
      FITFLAG = 0
      GO TO 90909
      ENDIF
      U(IH) = (WIND(1)*H(2,2) - WIND(2)*H(1,2))/DET
      V(IH) = (H(1,1)*WIND(2) - H(1,2)*WIND(1))/DET
C  CALCULATE THE STANDARD DEVIATION (SIGMA)
      ERRORSUM = 0
      DO 10401 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10401
```

```
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1           - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      ERRORSUM = ERRORSUM + DVR(POINT)**2
10401 CONTINUE
      SIGMA = SQRT(ERRORSUM/NPH)
      DO 10501 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10501
      IF (ABS(DVR(POINT)) .GT. NSIGMA*SIGMA) THEN
      IWT(POINT) = 0
      FLAG = 1
      NPH = NPH - 1
      IF (NPH .LT. MINH) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      ENDIF
10501 CONTINUE
      IF (FLAG .EQ. 0) GO TO 20002
      IF (FLAG .EQ. 1) THEN
      IF (SIGMA .GE. 0.999*SIGMALAST) GO TO 20002
      IF (SIGMA .LE. 0.01) GO TO 20002
      SIGMALAST = SIGMA
      GO TO 20001
      ENDIF
C  GOOD VELOCITY.
20002 IF ( (ABS(U(IH)) .GT. VMAX)      .OR.
     1     (ABS(V(IH)) .GT. VMAX)      .OR.
     2     (ABS(W(IH)) .GT. VMAX/20) ) THEN
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (FITFLAG .EQ. 1) THEN
      RMSDVR(IH) = 0
      DO 10601 POINT = 1,NPOINTS(IH)
      IF (IWT(POINT) .EQ. 0) GO TO 10601
      DVR(POINT) = SPP(POINT,2) - U(IH)*COSL(POINT)
     1           - V(IH)*COSM(POINT) - W(IH)*COSN(POINT)
      RMSDVR(IH) = RMSDVR(IH) + DVR(POINT)**2
      ZA = (180/PI)*ASIN(SQRT(1-COSN(POINT)**2))
      IZA = INT(ZA) + 1
      IF (IZA .GT. 17) THEN
      WRITE (*,*) 'IZA = ',IZA
      FITFLAG = 0
      GO TO 90909
      ENDIF
      IF (IZA .EQ. 17) IZA = 16
      VRAD(IZA) = VRAD(IZA) + DVR(POINT)**2
      NUMRAD(IZA) = NUMRAD(IZA) + 1
10601 CONTINUE
      IF (NPH .GT. 0) THEN
      RMSDVR(IH) = SQRT(RMSDVR(IH)/NPH)
      DO 10701 IALPHA = 1,16
      IF (NUMRAD(IALPHA) .EQ. 0) GO TO 10701
C     WRITE (*,*) 'ZA,VRAD,NUMRAD= ',IALPHA,VRAD(IALPHA),NUMRAD(IALPHA)
      VRAD(IALPHA) = SQRT(VRAD(IALPHA)/NUMRAD(IALPHA))
10701 CONTINUE
      ENDIF
      ENDIF
```

```
90909 RETURN
      END
      SUBROUTINE PHFIT
************************************************************************
C
C   THIS SUBROUTINE FITS A STRAIGHT LINE TO THE VARIATION OF VELOCITY
C   VARIANCE VS ZENITH ANGLE.
C   JULY 23, 1990
C
************************************************************************
      CHARACTER*1 POLAR
      CHARACTER*40 INFILE,OUTFILE
      CHARACTER*27 INPATH
      CHARACTER*19 OUTPATH
      CHARACTER*10 TUREKTIME
      INTEGER*4 YEAR,MONTH,DAY,HOUR,MINUTE,
     1          REJ,IH,NPOINTS(50),HOWLONG,NPV,NPV0,TESTFLAG,FITFLAG,
     2          MINH,MINV,NUMRAD,NOW
      REAL*4 INTERCEPT
      COMMON /WIND1/ SPP(2300,7),SPPZ(15,2300,7)
      COMMON /WIND2/ Z,TRP(50),
     1             REJ(4),LINE(10),
     2             WIDTH(50),IWT(2300),RMSDVR(50),
     4             COSL(2300),COSM(2300),COSN(2300),DVR(2300),
     5             NUMRAD(17),VRAD(17)
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1            NSIGMA,TESTFLAG,IH,NPOINTS,INFILE,OUTFILE,
     2            INPATH,OUTPATH,NPH,NPV,NPV0,SLOPE,INTERCEPT,FITFLAG
      COMMON /SPPFILE/ IFILE,NFILE,POLAR
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
C
      SUMVR = 0
      SUMVRPH = 0
      SUMPH = 0
      SUMPH2 = 0
      SUMI = 0
      DO 10101 IALPHA = 1,17
      IF (NUMRAD(IALPHA) .EQ. 0) GO TO 10101
      ZA = IALPHA - 0.5
      SUMVR = SUMVR + VRAD(IALPHA)
      SUMVRPH = SUMVRPH + VRAD(IALPHA)*ZA
      SUMPH = SUMPH + ZA
      SUMPH2 = SUMPH2 + ZA**2
      SUMI = SUMI + 1
10101 CONTINUE
      IF (SUMI .GT. 0) THEN
      IF (SUMI*SUMPH2 - SUMPH**2 .GT. 0) THEN
      SLOPE = (SUMI*SUMVRPH - SUMVR*SUMPH)/(SUMI*SUMPH2 - SUMPH**2)
      INTERCEPT = (SUMVR - SLOPE*SUMPH)/SUMI
      ELSE
      SLOPE = 0
      INTERCEPT = 0
      ENDIF
      ENDIF
      RETURN
      END
      SUBROUTINE REORDER (U,V,W)
```

```
C
C     REORDERS IDI WIND OUTPUT FILES IN DESCENDING HEIGHT.
C     THIS PORTION OF THE PROGRAM IS SPECIFIC TO A 43KM HEIGHT RANGE
C
      CHARACTER*1 TAB,POLAR
      CHARACTER*10 TUREKTIME
      CHARACTER*40 INFILE,OUTFILE
      DIMENSION H(50),U(50),V(50),W(50),UI(50),VI(50),WI(50),TRP(50),
     *XH(50),RT(50),SL(50),ICPT(50)
      INTEGER*4 IFILE,YEAR,MONTH,DAY,HOUR,MINUTE,NPOINTS(50),HOWLONG
      COMMON /WIND3/ PI,VMAX,THMAXH,THMINH,THMAXV,THMINV,MINH,MINV,
     1             NSIGMA,TESTFLAG,IH,NPOINTS,INFILE,OUTFILE
      COMMON /SPPFILE/ IFILE,NFILE,POLAR
      COMMON /TIMER/ INTNUM,YEAR,MONTH,DAY,HOUR,MINUTE,TUREKTIME,
     *HOWLONG,NOW
      COMMON /ARRAYS/ A(43,33)
C
      TAB=CHAR(9)
      WRITE (*,*) " REORDERING FILES BY DESCENDING HEIGHTS"
      CLOSE (16)
      NERR=8
      OPEN (16,ERR=90950,FILE=OUTFILE,STATUS="OLD",FORM="FORMATTED")
      IH=1
90912 READ (16,90001,END=90920,) H(IH),UI(IH),VI(IH),WI(IH),TRP(IH),
     *XH(IH),RT(IH),SL(IH),ICPT(IH)
90001 FORMAT (9(E13.4))
      IH=IH+1
      GO TO 90912
90920 REWIND (16)
      L=0
      J=15
90921 I=J
      K=0
90922 WRITE (16,90002)  H(I),TAB,UI(I),TAB,VI(I),TAB,WI(I),TAB,TRP(I),
     *TAB,XH(I),TAB,RT(I),TAB,SL(I),TAB,ICPT(I)
90002 FORMAT (E13.4,8(A1,E13.4))
      L=L+1
      U(L)=UI(I)
      V(L)=VI(I)
      W(L)=WI(I)
      IF (I.EQ.1) GO TO 90940
      K=K+1
      IF (K.EQ.1) THEN
      I=I+28
      GO TO 90922
      END IF
      IF (K.EQ.2) THEN
      I=I-14
      GO TO 90922
      ELSE
      J=J-1
      GO TO 90921
      END IF
90940 CLOSE (16)
      RETURN
90950 WRITE (*,*) " ERROR IN REORDERING FILES. NERR = ",NERR
      PAUSE " CR TO EXIT"
      STOP
```

```
      END
      SUBROUTINE DEVIANT
      CHARACTER*1 C4H(4),BLANK,NEG
      CHARACTER*4 CH4,BLANK4,B(43,33)
      REAL NINES
C
      COMMON /ARRAYS/ A(43,33),B
      EQUIVALENCE (CH4,C4H)
C
C     CALCULATE IDI AND ISR COMPONENT DIFFERENCES, AND SET UP OUTPUT
C     ARRAY FOR PRINTING AS XXXCRKOUT
C
      BLANK=CHAR(32)
      BLANK4="    "
      NINES=999.0
      NEG="-"
      DO 40 I=1,43
      IF (A(I,8).EQ.NINES)  GO TO 40
      IF (A(I,10).EQ.NINES) GO TO 40
      IF (A(I,14).EQ.NINES)  GO TO 40
      IF (A(I,16).EQ.NINES) GO TO 40
      A(I,18)=ABS(A(I,8)-A(I,14))
      A(I,19)=ABS(A(I,10)-A(I,16))
   40 CONTINUE
C
C     SET UP CHARACTER ARRAY B(43,33) FOR PRINTING
C
      DO 45 I=1,43
      DO 45 J=1,33
      IF (A(I,J).EQ.NINES) THEN
      CH4=BLANK
      GO TO 44
      END IF
      SIGN=A(I,J)/ABS(A(I,J))
      IA=ABS(A(I,J))
      IF (IA.EQ.0) A(I,J)=A(I,J)+1.0
      IA100=IA/100
      IA10=IA/10-10*IA100
      IA1=IA-100*IA100-10*IA10
      C4H(1)=BLANK
      C4H(2)=CHAR(IA100+48)
      IF (IA100.EQ.0) C4H(2)=BLANK
      C4H(3)=CHAR(IA10+48)
      C4H(4)=CHAR(IA1+48)
      IF (J.EQ.1) GO TO 42
      IF (C4H(2).NE.BLANK) GO TO 42
      IF (IA10.EQ.0) C4H(3)=BLANK
   42 IF (SIGN.GT.0.0) GO TO 44
      DO 43 K=3,1,-1
      IF (C4H(K).NE.BLANK) GO TO 43
      C4H(K)=NEG
      GO TO 44
   43 CONTINUE
   44 B(I,J)=CH4
   45 CONTINUE
      RETURN
      END
```

## The FPS and IDI wind comparison program FPSREDUCT

This program prepares a tab spaced column output file of zonal and meridional greenline profile smoothed a) hourly mean GROVES winds and b) IDI winds (created by IDIWIND.f), and Fabry-Perot spectrometer winds, suitable for reading into a plotting program.

INPUT

Reads either XXXXXXTIDE or XXXXXXGROOUT file output from GROVES, and smooths with a greenline profile to produce zonal, meridional and vertical winds at hourly intervals from 1900 LMST to 0500 LMST for the night of interest. Accesses the "TIDE" file if GROVES output consists only of mean (prevailing wind), diurnal and semidiurnal components only, via SUBROUTINE GLINE2, or accesses the "GROOUT" file (read by SUBROUTINE GLINE!) if more than two harmonic periods have been generated by GROVES. Selection is made by entering "T" (for "TIDE") or "G" (for "GROOUT") at the prompt request.

Enter FPS input (on disc) and output (your choice) filenames when prompted.

Enter IDI input file XXXXXXXX.MAW height spacing (all .MAW files generated after March 1, 1993, have 1km height spacing. Most of those generated before this date are spaced 3km). All files created by IDIWIND.f are 1km spacing.

Reads file SET.TIME, which is simply a listing of all the interval center point times of the .MAW files to be accessed, formatted as follows

```
8905031215
8905031322
..........
..........
..........
0000000000        ! EOF FLAG
```

Program then uses SUBROUTINE INNAME to create the appropriate IDI input filename.

After accessing the selected .MAW file, SUBROUTINE GREENIDI is then called, and the IDI profiles are greenline smoothed, and the ASCII output file written.

```
C     PROGRAM FPSREDUCT
C                                     APRIL 20, 1993
C     PREPARES CRICKETGRAPH FILE FOR COMPARISON OF
C     GROVES, IDI AND FABRY-PEROT WINDS
C     TO DETERMINE GROVES, READS EITHER XXXXXXGROOUT
C     (WHICH MAY HAVE BEEN GENERATED USING 2, 3 OR 4 HARMONICS
C     OR READS XXXXXXTIDE IF ONLY 2 HARMONICS WERE GENERATED.
C
      CHARACTER*1 DUM(8),TAB,WHICH
      CHARACTER 5 BLANK5
      CHARACTER*6 DUMMY1,DUMMY2
      CHARACTER*40 FILEIN,FILEOUT,GROOUT,TIDE,MAWFILE
      DIMENSION TIME(72),VEW(72),FERREW(72),VNS(72),FERRNS(72)
      INTEGER*4 YEAR,DAY,HOUR,MINUTE
      REAL*4 NINES
      COMMON /IDI/ PULSE,VEWIDI(72),ERREW(72),VNSIDI(72),ERRNS(72),
     *VWIDI(72),ERRW(72)
      COMMON /IDIG/ GTIME(24),GREW(24),GRNS(24),GRW(24)
      EQUIVALENCE (GROOUT,DUMMY1)
      EQUIVALENCE (TIDE,DUMMY2)
      EQUIVALENCE (MAWFILE,DUM)
      COMMON /IDIFILES/ MAWFILE,MONTH,DAY,HOUR,MINUTE
      TAB=CHAR(9)
      NINES=999.0
      DO 20 N=1,24
      GTIME(N)=NINES
      VEWIDI(N)=NINES
   20 CONTINUE
      DO 21 N=1,72
      TIME(N)=NINES
      VEW(N)=NINES
      VNS(N)=NINES
   21 CONTINUE
      BLANK5="     "
C
      WRITE (*,*) " PROGRAM FPSREDUCT"
      WRITE (*,*) " "
C
C     GET GROVES DATA  (XXXXGROOUT OR XXXXXXTIDE FILE)
C
      WRITE (*,*) " ENTER T FOR TIDE, G FOR GROOUT INPUT"
      READ (*,*) WHICH
      TIDE="      TIDE"
      GROOUT="      GROOUT"
      WRITE (*,*) " ENTER TIDE/GROOUT FILENUMBER"
      READ (*,*) DUMMY1
C
      IF (WHICH.EQ."G") THEN
      OPEN (17,FILE=GROOUT,ACTION="READ",FORM="FORMATTED")
      CALL GLINE1
      ELSE
      DUMMY2=DUMMY1
      OPEN (17,FILE=TIDE,ACTION="READ",FORM="UNFORMATTED")
      CALL GLINE2
      END IF
      CLOSE (17)
C
C     GET FPI DATA    (FPIYYYYY FILE)
C
      WRITE (*,*) " ENTER FPI INPUT, OUTPUT FILENAMES"
      READ (*,*) FILEIN,FILEOUT
      OPEN (17,FILE=FILEIN,ACTION="READ",FORM="FORMATTED")
      OPEN (16,FILE=FILEOUT,FORM="FORMATTED")
```

```
      VEL=50.0/24.0
      DO 10 I=1,72
      READ (17,*,END=11) VEW(I),FERREW(I),VNS(I),FERRNS(I)
      VEW(I)=VEL*VEW(I)
      VNS(I)=VEL*VNS(I)
      FERREW(I)=VEL*FERREW(I)
      FERRNS(I)=VEL*FERRNS(I)
      WRITE (*,*) VEW(I),FERREW(I),VNS(I),FERRNS(I)
   10 CONTINUE
   11 TIME(I)=NINES
      CLOSE (17)
C
C     GET IDI WINDS (MOJOHRMIN.MAW FILES)
C
      WRITE (*,*) " ENTER IDI HEIGHT SPACING - 1 OR 3 KM"
      READ (*,*) LPULSE
      PULSE=0.2
      IF (LPULSE.EQ.3) PULSE=0.08
      WRITE (*,*) " PULSE = ",PULSE
      OPEN (27,FILE="SET.TIME",ACTION="READ",FORM="FORMATTED")
      I=1
   12 READ (27,101,END=13) YEAR,MONTH,DAY,HOUR,MINUTE
  101 FORMAT (5I2)
      IF (YEAR.EQ.0) GO TO 13
      TIME(I)=FLOAT(HOUR)+FLOAT(MINUTE)/60.0
      IF (TIME(I).LT.12.0) TIME(I)=TIME(I)+24.0
      CALL INNAME
      WRITE (*,*) " PROCESSING FILE ",MAWFILE
      OPEN (17,FILE=MAWFILE,STATUS="OLD",ACTION="READ",FORM="FORMATTED")
      CALL GREENIDI (I)
      CLOSE (17)
C
C     GET NEXT IDI FILE
C
      I=I+1
      GO TO 12
C
C     PREPARE OUTPUT FILE
C
   13 CLOSE (17)
      CLOSE (27)
      J=1
      K=1
   14 WRITE (*,*) GTIME(J),TIME(K)
      IF (GTIME(J).EQ.NINES.AND.TIME(K).EQ.NINES) GO TO 15
      IF (GTIME(J).LT.TIME(K)) THEN
      WRITE (16,102) GTIME(J),TAB,GREW(J),TAB,GRNS(J),TAB,GRW(J)
  102 FORMAT (F8.3,3(A1,F8.0))
      J=J+1
      GO TO 14
      ELSE
      IF (VEWIDI(K).LT.900.0.AND.VNSIDI(K).LT.900.0
     *.AND.VEW(K).LT.900.0.AND.VNS(K).LT.900.0)       WRITE (16,103)
     *TIME(K),TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,TAB,VEWIDI(K),TAB,
     *ERREW(K),TAB,VNSIDI(K),TAB,ERRNS(K),TAB,VWIDI(K),TAB,ERRW(K),
     *TAB,VEW(K),TAB,FERREW(K), TAB,VNS(K),TAB,FERRNS(K)
  103 FORMAT (F8.3,3(A1,A5),10(A1,F8.0))
      IF (VEWIDI(K).GT.900.0.OR.VNSIDI(K).GT.900.0
     *.AND.VEW(K).LT.900.0.AND.VNS(K).LT.900.0)       WRITE (16,104)
     *TIME(K),TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,
     *TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,TAB,VEW(K),TAB,
     *FERREW(K),TAB,VNS(K),TAB,FERRNS(K)
  104 FORMAT (F8.3,9(A1,A5),4(A1,F8.0))
      IF (VEWIDI(K).LT.900.0.AND.VNSIDI(K).LT.900.0
```

```
     *.AND.VEW(K).GT.9....OR.VNS(K).GT.9...) THEN
      WRITE (16,105) TIME(K),TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,TAB,
     *VEWIDI(K),TAB,ERREW(K),TAB,VNSIDI(K),TAB,ERRNS(K),TAB,VNIDI(K),
     *TAB,ERRW(K),TAB,BLANK5,TAB,BLANK5,TAB,BLANK5,TAB,BLANK5
  105 FORMAT (F8.3,3(A1,A5),6(A1,F8.3),4(A1,A5))
      END IF
      K=K+1
      GO TO 14
      END IF
   15 CLOSE (16)
      PAUSE " ALL DONE - CR TO EXIT"
      STOP
      END
      SUBROUTINE GLINE1
C                                                  MARCH 8, 1993
C     READS GROVES WIND FROM FILE "XXXXXXGROOUT"
C
      CHARACTER*8 DIRNW,WHATW
          CHARACTER*9 DIRNEW,WHATEW
      CHARACTER*11 DIRNNS,WHATNS
      DIMENSION VX(24),VY(24),VZ(24),WT(10)
      COMMON /IDIG/ GTIME (24),GREW(24),GRNS(24),GRW(24)
      DATA WT/0.04867,0.07028,0.09352,0.1147,0.1296,0.1350,
     *0.1296,0.1147,0.09352,0.07028/
      DO 10 I=1,11
      GREW(I)=0.0
      GRNS(I)=0.0
      GTIME(I)=18.0+FLOAT(I)
   10 CONTINUE
      DIRNEW="EAST-WEST"
      DIRNNS="NORTH-SOUTH"
          DIRNW="VERTICAL"
C
C     SKIP DOWN TO EAST-WEST 102KM
C
    1 READ (17,*) WHATEW
      IF (WHATEW.NE.DIRNEW) GO TO 1
      DO 2 I=1,16
      READ (17,*) WHATEW
    2 CONTINUE
C
C     DETERMINE GREEN LINE SMOOTHED VELOCITY
C
      DO 3 K=1,10
      READ (17,100) IH,(VX(J),J=1,11)
      WRITE (*,100) IH,(VX(J),J=1,11)
  100 FORMAT (I5,38X,11F5.0)
      DO 3 I=1,11
      GREW(I)=GREW(I)+VX(I)*WT(K)
    3 CONTINUE
C
C     SKIP DOWN TO NORTH - SOUTH 102KM
C
    4 READ (17,*) WHATNS
      IF (WHATNS.NE.DIRNNS) GO TO 4
      DO 5 I=1,16
      READ (17,*) WHATNS
    5 CONTINUE
C
C     DETERMINE GREEN LINE SMOOTHED VELOCITY
C
      DO 6 K=1,10
      READ (17,100) IH,(VY(J),J=1,11)
      WRITE (*,100) IH,(VY(J),J=1,11)
```

```
      DO 6 I=1,11
      GRNS(I)=(GRNS(I)+VY(I)*WT(K))
    6 CONTINUE
C
C     SKIP DOWN TO VERTICAL 102KM
C
    7 READ (17,*) WHATW
      IF (WHATNS.NE.DIRNW) GO TO 7
      DO 8 I=1,16
      READ (17,*) WHATW
    8 CONTINUE
C
C     DETERMINE GREEN LINE SMOOTHED VELOCITY
C
      DO 9 K=1,10
      READ (17,100) IH,(VZ(J),J=1,11)
      WRITE (*,100) IH,(VZ(J),J=1,11)
      DO 9 I=1,11
      GRW(I)=(GRW(I)+VZ(I)*WT(K))
    9 CONTINUE
C
      RETURN
      END
      SUBROUTINE GLINE2
C                                                MARCH 6, 1993
C     CALCULATES TIDAL WIND FROM TIDAL COEFFICIENTS,
C     READING FILE "XXXXXXTIDE"
C
      DIMENSION AU(4),PH(4),VX(10,24),VY(10,24),VZ(10,24),WT(10)
      COMMON /IDIG/ GTIME (24),GREW(24),GRNS(24),GRW(24)
      DATA WT/0.04867,0.07028,0.09352,0.1147,0.1296,0.1350,
     *0.1296,0.1147,0.09352,0.07028/
      TWOPI=6.28318
      DO 10 I=1,11
      GREW(I)=0.0
      GRNS(I)=0.0
      GTIME(I)=18.0+FLOAT(I)
   10 CONTINUE
C
C     SKIP DOWN TO EAST-WEST 102KM
C
      DO 1 I=1,14
      READ (17) UO,(AU(J),PH(J),J=1,2)
    1 CONTINUE
C
C     DETERMINE GREEN LINE SMOOTHED VELOCITY
C
      DO 2 I=1,10
      READ (17) UO,(AU(J),PH(J),J=1,2)
      DO 2 IT=1,11
      T=GTIME(IT)
      IF (PH(1).LT.T) PHASE1=PH(1)-T
      IF (PH(1).GE.T) PHASE1=T-PH(1)
      IF (PH(2).LT.T) PHASE2=PH(2)-T
      IF (PH(2).GE.T) PHASE2=T-PH(2)
      VX(I,IT)=UO+AU(1)*COS((PHASE1/24.0)*TWOPI)
     *+AU(2)*COS((PHASE2/12.0)*TWOPI)
    2 CONTINUE
      DO 3 J=1,11
      DO 3 I=1,10
      GREW(J)=GREW(J)+VX(I,J)*WT(I)
    3 CONTINUE
C
C     SKIP DOWN TO NORTH - SOUTH 102KM
```

```
C
      DO 4 I=1,41
      READ (17) UO,(AU(J),PH(J),J=1,2)
    4 CONTINUE

C
C     DETERMINE GREEN LINE SMOOTHED VELOCITY
C
      DO 5 I=1,10
      READ (17) UO,(AU(J),PH(J),J=1,2)
      DO 5 IT=1,11
      T=GTIME(IT)
      IF (PH(1).LT.T) PHASE1=PH(1)-T
      IF (PH(1).GE.T) PHASE1=T-PH(1)
      IF (PH(2).LT.T) PHASE2=PH(2)-T
      IF (PH(2).GE.T) PHASE2=T-PH(2)
      VY(I,IT)=UO+AU(1)*COS((PHASE1/24.0)*TWOPI)
     *+AU(2)*COS((PHASE2/12.0)*TWOPI)
    5 CONTINUE
      DO 6 J=1,11
      DO 6 I=1,10
      GRNS(J)=GRNS(J)+VY(I,J)*WT(I)
    6 CONTINUE
C
C     SKIP DOWN TO VERTICAL 102KM
C
      DO 7 I=1,42
      READ (17) UO,(AU(J),PH(J),J=1,2)
    7 CONTINUE

C
C     DETERMINE GREEN LINE SMOOTHED VELOCITY
C
      DO 8 I=1,10
      READ (17) UO,(AU(J),PH(J),J=1,2)
      DO 8 IT=1,11
      T=GTIME(IT)
      IF (PH(1).LT.T) PHASE1=PH(1)-T
      IF (PH(1).GE.T) PHASE1=T-PH(1)
      IF (PH(2).LT.T) PHASE2=PH(2)-T
      IF (PH(2).GE.T) PHASE2=T-PH(2)
      VZ(I,IT)=UO+AU(1)*COS((PHASE1/24.0)*TWOPI)
     *+AU(2)*COS((PHASE2/12.0)*TWOPI)
    8 CONTINUE
      DO 9 J=1,11
      DO 9 I=1,10
      GRW(J)=(GRW(J)+VZ(I,J)*WT(I))
    9 CONTINUE
      RETURN
      END
      SUBROUTINE INNAME
C
C     INNAME CREATES MOJOHRMIN.MAW INPUT FILENAMES.
C
      CHARACTER*2 ASCMONTH,ASCDAY,ASCHOUR,ASCMINUTE
      CHARACTER*40 MAWFILE
      CHARACTER*19 INPATH
      INTEGER*4 MONTH,DAY,HOUR,MINUTE
      COMMON /IDIFILES/ MAWFILE,MONTH,DAY,HOUR,MINUTE
C
      INPATH="MAXTOR600:OUTFILES:"
      IF (MONTH .LT. 10) THEN
      WRITE (ASCMONTH,90001) '0',MONTH
90001 FORMAT (A1,I1)
```

```
      ELSE
      WRITE (ASCMONTH,90002) MONTH
90002 FORMAT (I2)
      ENDIF
      IF (DAY .LT. 10) THEN
      WRITE (ASCDAY,90001) '0',DAY
      ELSE
      WRITE (ASCDAY,90002) DAY
      ENDIF
      IF (HOUR .LT. 10) THEN
      WRITE (ASCHOUR,90001) '0',HOUR
      ELSE
      WRITE (ASCHOUR,90002) HOUR
      ENDIF
      IF (MINUTE .LT. 10) THEN
      WRITE (ASCMINUTE,90001) '0',MINUTE
      ELSE
      WRITE (ASCMINUTE,90002) MINUTE
      ENDIF
      WRITE (MAWFILE,90003)
     1 INPATH,ASCMONTH,ASCDAY,ASCHOUR,ASCMINUTE,'.MAW'
90003 FORMAT (A19,4A2,A4)
      RETURN
      END
      SUBROUTINE GREENIDI (I)
      DIMENSION VX(10),VY(10),VZ(10),WT(10)
      REAL*4 NINES
      COMMON /IDI/ PULSE,VEWIDI(72),ERREW(72),VNSIDI(72),ERRNS(72),
     *VWIDI(72),ERRW(72)
      DATA WT/0.04867,0.07028,0.09352,0.1147,0.1296,0.1350,
     *0.1296,0.1147,0.09352,0.07028/
      NINES=999.0
      VXSUM=0.0
      VYSUM=0.0
      VZSUM=0.0
      WTSUM=0.0
      ONE=1.0
      J=1
    3 READ (17,"(4E13.4)",END=4) Z,VX(J),VY(J),VZ(J)
      IDIZ=Z
      IF (IDIZ.GT.102) GO TO 3
      IF (IDIZ.LT.93) GO TO 4
      IF (VX(J).GT.900.0) THEN
      J=J+1
      GO TO 3
      ELSE
      VXSUM=VXSUM+VX(J)*WT(J)
      VYSUM=VYSUM+VY(J)*WT(J)
      VZSUM=VZSUM+VZ(J)*WT(J)
      WTSUM=WTSUM+WT(J)
      J=J+1
      GO TO 3
      END IF
    4 IF (WTSUM.EQ.0.0) THEN
      VEWIDI(I)=NINES
      RETURN
      ELSE
      VEWIDI(I)=VXSUM*ONE/WTSUM
      VNSIDI(I)=VYSUM*ONE/WTSUM
      VWIDI(I)=VZSUM*100.0/WTSUM
      ERRWT=PULSE/WTSUM
      ERREW(I)=5.0+ABS(VEWIDI(I))*ERRWT
      ERRNS(I)=5.0+ABS(VNSIDI(I))*ERRWT
      ERRW(I)=100.0+ABS(VWIDI(I))*ERRWT
```

```
WRITE (*,*) VEWIDI(I),UNSIDI(I),UWIDI(I)
IF (ERREW(I).GT.40.0) THEN
VEWIDI(I)=NINES
ERREW(I)=NINES
END IF
IF (ERRNS(I).GT.40.0) THEN
UNSIDI(I)=NINES
ERRNS(I)=NINES
END IF
RETURN
END IF
END
```

## Appendix I

A listing of file TUREKFILE (accessed by ISRIDIIDIG.f) which gives the interval number, date, start time, end time and viewing azimuth of the Arecibo Observatory incoherent scatter radar for the three AIDA'89 campaigns. Because of problems with phase jitter in the MAPSTAR processor controller during the March Scene I campaign, only the Scene II and Scene III data have been tabulated.

| | | | | |
|---|---|---|---|---|
| 1 | 890329 | 60850 | 63653 | 393 |
| 2 | 890329 | 64102 | 70905 | 393 |
| 3 | 890329 | 71638 | 72834 | 303 |
| 4 | 890329 | 73534 | 74619 | 213 |
| 5 | 890329 | 75318 | 81825 | 123 |
| 6 | 890329 | 82523 | 83610 | 213 |
| 7 | 890329 | 84307 | 85352 | 303 |
| 8 | 890329 | 90052 | 92559 | 393 |
| 9 | 890329 | 92934 | 95440 | 393 |
| 10 | 890329 | 95729 | 102235 | 393 |
| 11 | 890329 | 102610 | 105117 | 393 |
| 12 | 890329 | 105816 | 110901 | 303 |
| 13 | 890329 | 111600 | 112646 | 213 |
| 14 | 890329 | 113345 | 115851 | 123 |
| 15 | 890329 | 120550 | 121635 | 213 |
| 16 | 890329 | 122333 | 123419 | 303 |
| 17 | 890329 | 124116 | 130622 | 393 |
| 18 | 890329 | 130959 | 133506 | 393 |
| 19 | 890329 | 133754 | 140300 | 393 |
| 20 | 890329 | 141730 | 150344 | 393 |
| 21 | 890329 | 151042 | 151316 | 303 |
| 22 | 890329 | 163938 | 170923 | 123 |
| 23 | 890329 | 171619 | 172703 | 213 |
| 24 | 890329 | 173401 | 174444 | 303 |
| 25 | 890329 | 175142 | 181641 | 393 |
| 26 | 890329 | 182013 | 184513 | 393 |
| 27 | 890329 | 184759 | 191306 | 393 |
| 28 | 890330 | 65640 | 72139 | 393 |
| 29 | 890330 | 72514 | 73557 | 393 |
| 30 | 890330 | 74253 | 75336 | 303 |
| 31 | 890330 | 80034 | 81117 | 213 |
| 32 | 890330 | 81814 | 84313 | 123 |
| 33 | 890330 | 85010 | 90053 | 213 |
| 34 | 890330 | 90750 | 91833 | 303 |
| 35 | 890330 | 92532 | 95039 | 393 |
| 36 | 890330 | 95415 | 101921 | 393 |
| 37 | 890330 | 102210 | 104717 | 393 |
| 38 | 890330 | 105053 | 111560 | 393 |
| 39 | 890330 | 112258 | 113345 | 303 |
| 40 | 890330 | 114042 | 115129 | 213 |
| 41 | 890330 | 115827 | 122334 | 123 |
| 42 | 890330 | 123033 | 124118 | 213 |
| 43 | 890330 | 124816 | 125902 | 303 |
| 44 | 890330 | 130602 | 133109 | 393 |
| 45 | 890330 | 133444 | 140320 | 393 |
| 46 | 890330 | 140609 | 143116 | 393 |
| 47 | 890330 | 143451 | 145959 | 393 |
| 48 | 890330 | 151243 | 152329 | 303 |
| 49 | 890330 | 153039 | 154124 | 213 |
| 50 | 890330 | 154824 | 161331 | 123 |
| 51 | 890330 | 162029 | 163115 | 213 |
| 52 | 890330 | 163812 | 164858 | 303 |
| 53 | 890330 | 165556 | 172103 | 393 |
| 54 | 890330 | 172438 | 174947 | 393 |

| | | | | |
|---|---|---|---|---|
| 55 | 890330 | 175235 | 181742 | 393 |
| 56 | 890330 | 182117 | 184624 | 393 |
| 57 | 890331 | 62312 | 64541 | 393 |
| 58 | 890331 | 64844 | 71114 | 393 |
| 59 | 890331 | 71811 | 72753 | 303 |
| 60 | 890331 | 73451 | 74434 | 213 |
| 61 | 890331 | 75131 | 81400 | 123 |
| 62 | 890331 | 82056 | 83040 | 213 |
| 63 | 890331 | 83736 | 84720 | 303 |
| 64 | 890331 | 85419 | 91648 | 393 |
| 65 | 890331 | 91951 | 94219 | 393 |
| 66 | 890331 | 94505 | 100738 | 393 |
| 67 | 890331 | 101041 | 103310 | 393 |
| 68 | 890331 | 104007 | 104950 | 303 |
| 69 | 890331 | 105648 | 110630 | 213 |
| 70 | 890331 | 111328 | 113558 | 123 |
| 71 | 890331 | 114254 | 115238 | 213 |
| 72 | 890331 | 115937 | 120919 | 303 |
| 73 | 890331 | 121618 | 123848 | 393 |
| 74 | 890331 | 124150 | 130420 | 393 |
| 75 | 890331 | 130706 | 132935 | 393 |
| 76 | 890331 | 133241 | 135514 | 393 |
| 77 | 890331 | 140212 | 141341 | 303 |
| 78 | 890331 | 142222 | 143349 | 213 |
| 79 | 890331 | 144229 | 151006 | 123 |
| 80 | 890331 | 151847 | 153012 | 213 |
| 81 | 890331 | 155001 | 160128 | 303 |
| 82 | 890331 | 161009 | 163756 | 393 |
| 83 | 890331 | 164244 | 171031 | 393 |
| 84 | 890331 | 171502 | 174248 | 393 |
| 85 | 890331 | 174736 | 181523 | 393 |
| 86 | 890331 | 182401 | 183530 | 303 |
| 87 | 890331 | 184411 | 185540 | 213 |
| 88 | 890331 | 190419 | 191548 | 123 |
| 89 | 890401 | 71634 | 74421 | 393 |
| 90 | 890401 | 74909 | 81657 | 393 |
| 91 | 890401 | 82538 | 83707 | 303 |
| 92 | 890401 | 84548 | 85717 | 213 |
| 93 | 890401 | 90559 | 93346 | 123 |
| 94 | 890401 | 94226 | 95356 | 213 |
| 95 | 890401 | 100237 | 101405 | 303 |
| 96 | 890401 | 102246 | 105033 | 393 |
| 97 | 890401 | 105521 | 112308 | 393 |
| 98 | 890401 | 112740 | 115526 | 393 |
| 99 | 890401 | 120015 | 122801 | 393 |
| 100 | 890401 | 123643 | 124812 | 303 |
| 101 | 890401 | 125654 | 130823 | 213 |
| 102 | 890401 | 131704 | 134452 | 123 |
| 103 | 890401 | 135333 | 140502 | 213 |
| 104 | 890401 | 141342 | 142511 | 303 |
| 105 | 890401 | 143352 | 150139 | 393 |
| 106 | 890401 | 150628 | 153416 | 393 |
| 107 | 890401 | 153847 | 160634 | 393 |
| 108 | 890401 | 161122 | 163910 | 393 |

| | | | | |
|---|---|---|---|---|
| 109 | 890401 | 164748 | 165917 | 303 |
| 110 | 890401 | 170759 | 171928 | 213 |
| 111 | 890401 | 172810 | 175556 | 123 |
| 112 | 890401 | 180436 | 181605 | 213 |
| 113 | 890401 | 182446 | 183615 | 303 |
| 114 | 890402 | 63617 | 65854 | 393 |
| 115 | 890402 | 70159 | 72436 | 393 |
| 116 | 890402 | 73134 | 74121 | 303 |
| 117 | 890402 | 74820 | 75806 | 213 |
| 118 | 890402 | 80504 | 82741 | 123 |
| 119 | 890402 | 83439 | 84424 | 213 |
| 120 | 890402 | 85121 | 90107 | 303 |
| 121 | 890402 | 90805 | 93041 | 393 |
| 122 | 890402 | 93346 | 95623 | 393 |
| 123 | 890402 | 95912 | 102148 | 393 |
| 124 | 890402 | 102453 | 104731 | 393 |
| 125 | 890402 | 105430 | 110415 | 303 |
| 126 | 890402 | 111114 | 112060 | 213 |
| 127 | 890402 | 112758 | 115035 | 123 |
| 128 | 890402 | 115733 | 120719 | 213 |
| 129 | 890402 | 121416 | 122402 | 303 |
| 130 | 890402 | 123060 | 125338 | 393 |
| 131 | 890402 | 125643 | 131921 | 393 |
| 132 | 890402 | 132209 | 134447 | 393 |
| 133 | 890402 | 134752 | 141030 | 393 |
| 134 | 890402 | 141729 | 142715 | 303 |
| 135 | 890402 | 143413 | 144360 | 213 |
| 136 | 890402 | 145057 | 151334 | 123 |
| 137 | 890402 | 152032 | 153018 | 213 |
| 138 | 890402 | 153715 | 154701 | 303 |
| 139 | 890402 | 155359 | 161637 | 393 |
| 140 | 890402 | 161942 | 164219 | 393 |
| 141 | 890402 | 164506 | 170744 | 393 |
| 142 | 890402 | 171049 | 173326 | 393 |
| 143 | 890402 | 174025 | 175011 | 303 |
| 144 | 890402 | 175709 | 180654 | 213 |
| 145 | 890402 | 181352 | 183629 | 123 |
| 146 | 890402 | 184326 | 185313 | 213 |
| 147 | 890403 | 60804 | 63041 | 393 |
| 148 | 890403 | 63346 | 65624 | 393 |
| 149 | 890403 | 70322 | 71308 | 303 |
| 150 | 890403 | 72007 | 72953 | 213 |
| 151 | 890403 | 73651 | 75929 | 123 |
| 152 | 890403 | 80629 | 81615 | 213 |
| 153 | 890403 | 82312 | 83258 | 303 |
| 154 | 890403 | 83958 | 90235 | 393 |
| 155 | 890403 | 90540 | 92817 | 393 |
| 156 | 890403 | 93105 | 95343 | 393 |
| 157 | 890403 | 95649 | 101926 | 393 |
| 158 | 890403 | 102623 | 103609 | 303 |
| 159 | 890403 | 104310 | 105255 | 213 |
| 160 | 890403 | 105953 | 112231 | 123 |
| 161 | 890403 | 113130 | 114115 | 213 |
| 162 | 890403 | 114813 | 115760 | 303 |

| | | | | |
|---|---|---|---|---|
| 163 | 890403 | 120504 | 122741 | 393 |
| 164 | 890403 | 123047 | 125324 | 393 |
| 165 | 890403 | 125612 | 131850 | 393 |
| 166 | 890403 | 132155 | 134433 | 393 |
| 167 | 890403 | 135131 | 140117 | 303 |
| 168 | 890403 | 140818 | 141804 | 213 |
| 169 | 890403 | 142502 | 144740 | 123 |
| 170 | 890403 | 145437 | 150423 | 213 |
| 171 | 890403 | 151120 | 152107 | 303 |
| 172 | 890403 | 152812 | 155050 | 393 |
| 173 | 890403 | 155355 | 161632 | 393 |
| 174 | 890403 | 161921 | 164158 | 393 |
| 175 | 890403 | 164503 | 170740 | 393 |
| 176 | 890403 | 171440 | 172426 | 303 |
| 177 | 890403 | 173124 | 174110 | 213 |
| 178 | 890403 | 174809 | 181047 | 123 |
| 179 | 890403 | 181745 | 182731 | 213 |
| 180 | 890403 | 183428 | 184414 | 303 |
| 181 | 890403 | 185113 | 191350 | 393 |
| 182 | 890403 | 191654 | 193933 | 393 |
| 183 | 890403 | 194221 | 200458 | 393 |
| 184 | 890403 | 200303 | 203040 | 393 |
| 185 | 890403 | 203739 | 204724 | 303 |
| 186 | 890403 | 205423 | 210409 | 213 |
| 187 | 890403 | 211107 | 213345 | 123 |
| 188 | 890403 | 214043 | 215029 | 213 |
| 189 | 890403 | 215727 | 220712 | 303 |
| 190 | 890403 | 221410 | 223646 | 393 |
| 191 | 890403 | 223952 | 224226 | 393 |
| 192 | 890404 | 73810 | 80557 | 123 |
| 193 | 890404 | 81440 | 82609 | 213 |
| 194 | 890404 | 83450 | 84619 | 303 |
| 195 | 890404 | 85500 | 92247 | 393 |
| 196 | 890404 | 92735 | 95523 | 393 |
| 197 | 890404 | 95954 | 103426 | 393 |
| 198 | 890404 | 103914 | 110660 | 393 |
| 199 | 890404 | 111542 | 112710 | 303 |
| 200 | 890404 | 113554 | 114722 | 213 |
| 201 | 890404 | 115604 | 122351 | 123 |
| 202 | 890404 | 123235 | 124404 | 213 |
| 203 | 890404 | 125245 | 130414 | 303 |
| 204 | 890404 | 131258 | 134044 | 393 |
| 205 | 890404 | 134533 | 141320 | 393 |
| 206 | 890404 | 141751 | 144537 | 393 |
| 207 | 890404 | 145026 | 151814 | 393 |
| 208 | 890404 | 152655 | 153823 | 303 |
| 209 | 890404 | 154706 | 155835 | 213 |
| 210 | 890404 | 160717 | 163504 | 123 |
| 211 | 890404 | 164345 | 165515 | 213 |
| 212 | 890404 | 170356 | 171525 | 303 |
| 213 | 890404 | 172405 | 175151 | 393 |
| 214 | 890404 | 175640 | 182426 | 393 |
| 215 | 890404 | 182857 | 184025 | 393 |
| 216 | 890405 | 64038 | 65024 | 303 |

| | | | | |
|---|---|---|---|---|
| 217 | 890405 | 65722 | 70708 | 213 |
| 218 | 890405 | 71407 | 73643 | 123 |
| 219 | 890405 | 74341 | 75327 | 213 |
| 220 | 890405 | 130318 | 132631 | 393 |
| 221 | 890405 | 132943 | 135253 | 393 |
| 222 | 890405 | 135958 | 140956 | 303 |
| 223 | 890405 | 141702 | 142701 | 213 |
| 224 | 890405 | 143406 | 145718 | 123 |
| 225 | 890405 | 150421 | 151420 | 213 |
| 226 | 890405 | 152122 | 153120 | 303 |
| 227 | 890405 | 153826 | 160135 | 393 |
| 228 | 890405 | 160447 | 162759 | 393 |
| 229 | 890405 | 163055 | 165405 | 393 |
| 230 | 890405 | 165716 | 172026 | 393 |
| 231 | 890405 | 172730 | 173729 | 303 |
| 232 | 890405 | 174434 | 175433 | 213 |
| 233 | 890405 | 180138 | 182451 | 123 |
| 234 | 890405 | 183154 | 184152 | 213 |
| 235 | 890406 | 64101 | 70412 | 393 |
| 236 | 890406 | 75943 | 82255 | 393 |
| 237 | 890406 | 82606 | 84915 | 393 |
| 238 | 890406 | 85619 | 90618 | 303 |
| 239 | 890406 | 91323 | 92322 | 213 |
| 240 | 890406 | 93027 | 95337 | 123 |
| 241 | 890406 | 100041 | 101039 | 213 |
| 242 | 890406 | 101742 | 102739 | 303 |
| 243 | 890406 | 103443 | 105751 | 393 |
| 244 | 890406 | 110103 | 112414 | 393 |
| 245 | 890406 | 112710 | 115019 | 393 |
| 246 | 890406 | 115331 | 121639 | 393 |
| 247 | 890406 | 122343 | 123342 | 303 |
| 248 | 890406 | 124046 | 125045 | 213 |
| 249 | 890406 | 125750 | 132102 | 123 |
| 250 | 890406 | 132807 | 133805 | 213 |
| 251 | 890406 | 134508 | 135505 | 303 |
| 252 | 890406 | 140209 | 142516 | 393 |
| 253 | 890406 | 142829 | 145139 | 393 |
| 254 | 890406 | 151152 | 152150 | 303 |
| 255 | 890406 | 152854 | 155201 | 393 |
| 256 | 890406 | 155514 | 161822 | 393 |
| 257 | 890406 | 162118 | 164429 | 393 |
| 258 | 890406 | 164741 | 171049 | 393 |
| 259 | 890406 | 171752 | 172750 | 303 |
| 260 | 890406 | 173452 | 174450 | 213 |
| 261 | 890406 | 175154 | 181504 | 123 |
| 262 | 890406 | 182206 | 182441 | 213 |
| 263 | 890406 | 184430 | 185428 | 303 |
| 264 | 890407 | 64800 | 80819 | 393 |
| 265 | 890407 | 81308 | 83420 | 393 |
| 266 | 890407 | 83911 | 84144 | 393 |
| 267 | 890407 | 84848 | 85848 | 303 |
| 268 | 890407 | 90551 | 91550 | 213 |
| 269 | 890407 | 92254 | 94602 | 123 |
| 270 | 890407 | 95306 | 100304 | 213 |

| | | | | |
|---|---|---|---|---|
| 271 | 890407 | 101008 | 102007 | 303 |
| 272 | 890407 | 102711 | 105023 | 393 |
| 273 | 890407 | 105336 | 111644 | 393 |
| 274 | 890407 | 111939 | 114247 | 393 |
| 275 | 890407 | 114600 | 120913 | 393 |
| 276 | 890407 | 121618 | 122618 | 303 |
| 277 | 890407 | 123321 | 124321 | 213 |
| 278 | 890407 | 125024 | 131333 | 123 |
| 279 | 890407 | 132037 | 133036 | 213 |
| 280 | 890407 | 133739 | 134737 | 303 |
| 281 | 890407 | 135442 | 141745 | 393 |
| 282 | 890407 | 142059 | 144408 | 393 |
| 283 | 890407 | 144703 | 151011 | 393 |
| 284 | 890407 | 151323 | 153634 | 393 |
| 285 | 890407 | 154338 | 155338 | 303 |
| 286 | 890407 | 160043 | 161042 | 213 |
| 287 | 890407 | 161746 | 164056 | 123 |
| 288 | 890407 | 164759 | 165757 | 213 |
| 289 | 890407 | 170460 | 171458 | 303 |
| 290 | 890407 | 172202 | 174511 | 393 |
| 291 | 890407 | 174824 | 181136 | 393 |
| 292 | 890407 | 181430 | 183738 | 393 |
| 293 | 890407 | 184049 | 190359 | 393 |
| 294 | 890407 | 191103 | 192103 | 303 |
| 295 | 890408 | 63835 | 70145 | 123 |
| 296 | 890408 | 70847 | 71846 | 213 |
| 297 | 890408 | 72549 | 73546 | 303 |
| 298 | 890408 | 74250 | 80560 | 393 |
| 299 | 890408 | 80914 | 83229 | 393 |
| 300 | 890408 | 83523 | 85831 | 393 |
| 301 | 890408 | 90143 | 92454 | 393 |
| 302 | 890408 | 93158 | 94157 | 303 |
| 303 | 890408 | 94902 | 95901 | 213 |
| 304 | 890408 | 100607 | 102919 | 123 |
| 305 | 890408 | 103622 | 104620 | 213 |
| 306 | 890408 | 105322 | 110321 | 303 |
| 307 | 890408 | 111025 | 113334 | 393 |
| 308 | 890408 | 113646 | 115958 | 393 |
| 309 | 890408 | 120254 | 122601 | 393 |
| 310 | 890408 | 122914 | 125222 | 393 |
| 311 | 890408 | 125926 | 130925 | 303 |
| 312 | 890408 | 131630 | 132630 | 213 |
| 313 | 890408 | 133334 | 135646 | 123 |
| 314 | 890408 | 140350 | 141350 | 213 |
| 315 | 890408 | 142053 | 143051 | 303 |
| 316 | 890408 | 143754 | 150102 | 393 |
| 317 | 890408 | 150414 | 152726 | 393 |
| 318 | 890408 | 153022 | 155606 | 393 |
| 319 | 890408 | 160056 | 162226 | 393 |
| 320 | 890408 | 162716 | 164302 | 393 |
| 321 | 890408 | 165007 | 170007 | 303 |
| 322 | 890408 | 170711 | 171711 | 213 |
| 323 | 890408 | 172416 | 174726 | 123 |
| 324 | 890408 | 175427 | 180426 | 213 |

| | | | | |
|---|---|---|---|---|
| 325 | 890408 | 181129 | 182127 | 303 |
| 326 | 890408 | 182828 | 185138 | 393 |
| 327 | 890408 | 185452 | 191803 | 393 |
| 328 | 890408 | 192058 | 194559 | 393 |
| 329 | 890409 | 63251 | 65422 | 393 |
| 330 | 890409 | 65912 | 70146 | 393 |
| 331 | 890409 | 70849 | 71848 | 303 |
| 332 | 890409 | 72551 | 73550 | 213 |
| 333 | 890409 | 74254 | 80604 | 123 |
| 334 | 890409 | 81309 | 82309 | 213 |
| 335 | 890409 | 83014 | 84013 | 303 |
| 336 | 890409 | 84717 | 91026 | 393 |
| 337 | 890409 | 91338 | 93645 | 393 |
| 338 | 890409 | 93940 | 100252 | 393 |
| 339 | 890409 | 100605 | 102913 | 393 |
| 340 | 890409 | 103617 | 104615 | 303 |
| 341 | 890409 | 105319 | 110317 | 213 |
| 342 | 890409 | 111020 | 113330 | 123 |
| 343 | 890409 | 114034 | 115034 | 213 |
| 344 | 890409 | 115739 | 120738 | 303 |
| 345 | 890409 | 121443 | 123754 | 393 |
| 346 | 890409 | 124106 | 130414 | 393 |
| 347 | 890409 | 130708 | 133018 | 393 |
| 348 | 890409 | 133332 | 135642 | 393 |
| 349 | 890409 | 140346 | 141344 | 303 |
| 350 | 890409 | 142048 | 143046 | 213 |
| 351 | 890409 | 143749 | 150058 | 123 |
| 352 | 890409 | 150802 | 151800 | 213 |
| 353 | 890409 | 152504 | 153504 | 303 |
| 354 | 890409 | 154208 | 160519 | 393 |
| 355 | 890409 | 160833 | 163140 | 393 |
| 356 | 890409 | 163435 | 165744 | 393 |
| 357 | 890409 | 170058 | 172411 | 393 |
| 358 | 890409 | 173114 | 174114 | 303 |
| 359 | 890409 | 174818 | 175816 | 213 |
| 360 | 890409 | 180519 | 182828 | 123 |
| 361 | 890409 | 183531 | 184530 | 213 |
| 362 | 890409 | 185233 | 190232 | 303 |
| 363 | 890410 | 60817 | 63128 | 393 |
| 364 | 890410 | 63441 | 65752 | 393 |
| 365 | 890410 | 70456 | 71454 | 303 |
| 366 | 890410 | 72158 | 73157 | 213 |
| 367 | 890410 | 73901 | 80210 | 123 |
| 368 | 890410 | 80915 | 81914 | 213 |
| 369 | 890410 | 82616 | 83616 | 303 |
| 370 | 890410 | 84322 | 90634 | 393 |
| 371 | 890410 | 91633 | 93944 | 393 |
| 372 | 890410 | 94240 | 100548 | 393 |
| 373 | 890410 | 100900 | 103209 | 393 |
| 374 | 890410 | 103913 | 104912 | 303 |
| 375 | 890410 | 105616 | 110616 | 213 |
| 376 | 890410 | 111320 | 113634 | 123 |
| 377 | 890410 | 114342 | 115340 | 213 |
| 378 | 890410 | 120043 | 121041 | 303 |

| | | | | |
|---|---|---|---|---|
| 379 | 890410 | 121745 | 124054 | 393 |
| 380 | 890410 | 124405 | 130716 | 393 |
| 381 | 890410 | 131012 | 133322 | 393 |
| 382 | 890410 | 133634 | 135942 | 393 |
| 383 | 890410 | 140648 | 141646 | 303 |
| 384 | 890410 | 142350 | 143349 | 213 |
| 385 | 890410 | 144054 | 150406 | 123 |
| 386 | 890410 | 151109 | 152109 | 213 |
| 387 | 890410 | 153536 | 153810 | 303 |
| 388 | 890410 | 154730 | 161038 | 393 |
| 389 | 890410 | 161351 | 163700 | 393 |
| 390 | 890410 | 163955 | 170307 | 393 |
| 391 | 890410 | 170619 | 172927 | 393 |
| 392 | 890410 | 173631 | 174630 | 303 |
| 393 | 890410 | 175333 | 180332 | 213 |
| 394 | 890410 | 181036 | 183347 | 123 |
| 395 | 890410 | 184052 | 185051 | 213 |
| 396 | 890411 | 61421 | 64026 | 393 |
| 397 | 890411 | 64413 | 71019 | 393 |
| 398 | 890411 | 71758 | 72907 | 303 |
| 399 | 890411 | 73645 | 74755 | 213 |
| 400 | 890411 | 75533 | 82140 | 123 |
| 401 | 890411 | 82917 | 84027 | 213 |
| 402 | 890411 | 84805 | 85914 | 303 |
| 403 | 890411 | 90652 | 93258 | 393 |
| 404 | 890411 | 93644 | 105404 | 393 |
| 405 | 890411 | 105927 | 112513 | 393 |
| 406 | 890411 | 113002 | 113821 | 393 |
| 501 | 890501 | 172508 | 174344 | 393 |
| 502 | 890501 | 174558 | 181050 | 393 |
| 503 | 890501 | 181305 | 190917 | 393 |
| 504 | 890502 | 61539 | 64059 | 393 |
| 505 | 890502 | 64325 | 70846 | 393 |
| 506 | 890502 | 71457 | 72624 | 303 |
| 507 | 890502 | 73236 | 74403 | 213 |
| 508 | 890502 | 75015 | 75250 | 123 |
| 509 | 890502 | 81232 | 82402 | 213 |
| 510 | 890502 | 83019 | 85546 | 123 |
| 511 | 890502 | 90602 | 91731 | 213 |
| 512 | 890502 | 93402 | 94532 | 303 |
| 513 | 890502 | 95451 | 102018 | 393 |
| 514 | 890502 | 102245 | 105536 | 393 |
| 515 | 890502 | 105735 | 112301 | 393 |
| 516 | 890502 | 112529 | 115055 | 393 |
| 517 | 890502 | 115712 | 120840 | 303 |
| 518 | 890502 | 121457 | 122627 | 213 |
| 519 | 890502 | 123243 | 125809 | 123 |
| 520 | 890502 | 130426 | 131556 | 213 |
| 521 | 890502 | 132212 | 133342 | 303 |
| 522 | 890502 | 133958 | 145746 | 393 |
| 523 | 890502 | 150014 | 152541 | 393 |
| 524 | 890502 | 152740 | 155306 | 393 |
| 525 | 890502 | 155534 | 162101 | 393 |
| 526 | 890502 | 162718 | 163848 | 303 |

| | | | | |
|---|---|---|---|---|
| 527 | 890502 | 164504 | 165634 | 213 |
| 528 | 890502 | 170250 | 172817 | 123 |
| 529 | 890502 | 173433 | 174602 | 213 |
| 530 | 890502 | 175219 | 180349 | 303 |
| 531 | 890502 | 181005 | 183834 | 393 |
| 532 | 890503 | 64456 | 70628 | 393 |
| 533 | 890503 | 71249 | 71524 | 393 |
| 534 | 890503 | 72140 | 73310 | 303 |
| 535 | 890503 | 73927 | 75056 | 213 |
| 536 | 890503 | 75712 | 82238 | 123 |
| 537 | 890503 | 82855 | 84025 | 213 |
| 538 | 890503 | 84641 | 85810 | 303 |
| 539 | 890503 | 90426 | 92953 | 393 |
| 540 | 890503 | 93220 | 95747 | 393 |
| 541 | 890503 | 95945 | 102512 | 393 |
| 542 | 890503 | 102740 | 105306 | 393 |
| 543 | 890503 | 105922 | 111051 | 303 |
| 544 | 890503 | 111707 | 112836 | 213 |
| 545 | 890503 | 113453 | 120019 | 123 |
| 546 | 890503 | 120635 | 121805 | 213 |
| 547 | 890503 | 122421 | 123551 | 303 |
| 548 | 890503 | 124208 | 130734 | 393 |
| 549 | 890503 | 131002 | 133528 | 393 |
| 550 | 890503 | 133726 | 140254 | 393 |
| 551 | 890503 | 140521 | 143047 | 393 |
| 552 | 890503 | 143704 | 144833 | 303 |
| 553 | 890503 | 145449 | 150618 | 213 |
| 554 | 890503 | 151444 | 154011 | 123 |
| 555 | 890503 | 154628 | 155757 | 213 |
| 556 | 890503 | 160414 | 161543 | 303 |
| 557 | 890503 | 162160 | 164726 | 393 |
| 558 | 890503 | 164953 | 171520 | 393 |
| 559 | 890503 | 171718 | 174245 | 393 |
| 560 | 890503 | 174513 | 181039 | 393 |
| 561 | 890503 | 181655 | 182825 | 303 |
| 562 | 890504 | 61321 | 64040 | 393 |
| 563 | 890504 | 64308 | 71029 | 393 |
| 564 | 890504 | 71646 | 72912 | 303 |
| 565 | 890504 | 73528 | 74754 | 213 |
| 566 | 890504 | 75410 | 82131 | 123 |
| 567 | 890504 | 83024 | 84251 | 213 |
| 568 | 890504 | 84907 | 90134 | 303 |
| 569 | 890504 | 91038 | 93758 | 393 |
| 570 | 890504 | 94026 | 100746 | 393 |
| 571 | 890504 | 100946 | 103706 | 393 |
| 572 | 890504 | 103933 | 110654 | 393 |
| 573 | 890504 | 111310 | 112536 | 303 |
| 574 | 890504 | 113153 | 114420 | 213 |
| 575 | 890504 | 115036 | 121756 | 123 |
| 576 | 890504 | 122413 | 123639 | 213 |
| 577 | 890504 | 124256 | 125522 | 303 |
| 578 | 890504 | 130138 | 132858 | 393 |
| 579 | 890504 | 133127 | 135846 | 393 |
| 580 | 890504 | 140046 | 142806 | 393 |

| | | | | |
|---|---|---|---|---|
| 581 | 890504 | 143034 | 145754 | 393 |
| 582 | 890504 | 150411 | 151637 | 303 |
| 583 | 890504 | 152254 | 153520 | 213 |
| 584 | 890504 | 154137 | 160857 | 123 |
| 585 | 890504 | 161513 | 162739 | 213 |
| 586 | 890504 | 163356 | 164623 | 303 |
| 587 | 890504 | 165239 | 171959 | 393 |
| 588 | 890504 | 172227 | 174948 | 393 |
| 589 | 890504 | 175147 | 181906 | 393 |
| 590 | 890504 | 182134 | 184854 | 393 |
| 591 | 890505 | 62040 | 64607 | 393 |
| 592 | 890505 | 64834 | 71401 | 393 |
| 593 | 890505 | 72017 | 73146 | 303 |
| 594 | 890505 | 73802 | 74932 | 213 |
| 595 | 890505 | 75548 | 82114 | 123 |
| 596 | 890505 | 82730 | 83900 | 213 |
| 597 | 890505 | 84516 | 85646 | 303 |
| 598 | 890505 | 90302 | 92828 | 393 |
| 599 | 890505 | 93055 | 95622 | 393 |
| 600 | 890505 | 95821 | 102347 | 393 |
| 601 | 890505 | 102615 | 105141 | 393 |
| 602 | 890505 | 105758 | 110926 | 303 |
| 603 | 890505 | 111543 | 112712 | 213 |
| 604 | 890505 | 113328 | 115855 | 123 |
| 605 | 890505 | 120511 | 121641 | 213 |
| 606 | 890505 | 122257 | 123426 | 303 |
| 607 | 890505 | 124043 | 130609 | 393 |
| 608 | 890505 | 130837 | 133404 | 393 |
| 609 | 890505 | 133603 | 140129 | 393 |
| 610 | 890505 | 140357 | 142922 | 393 |
| 611 | 890505 | 143539 | 144708 | 303 |
| 612 | 890505 | 145325 | 150454 | 213 |
| 613 | 890505 | 151111 | 153637 | 123 |
| 614 | 890505 | 154254 | 155423 | 213 |
| 615 | 890505 | 160039 | 161209 | 303 |
| 616 | 890505 | 161825 | 164352 | 393 |
| 617 | 890505 | 164620 | 171146 | 393 |
| 618 | 890505 | 171344 | 173911 | 393 |
| 619 | 890505 | 174139 | 180705 | 393 |
| 620 | 890505 | 181322 | 182452 | 303 |
| 621 | 890505 | 183108 | 184237 | 213 |
| 622 | 890505 | 184853 | 191420 | 123 |
| 623 | 890505 | 192036 | 193205 | 213 |
| 624 | 890505 | 193822 | 194950 | 303 |
| 625 | 890505 | 195607 | 202133 | 393 |
| 626 | 890505 | 202400 | 204915 | 393 |
| 627 | 890506 | 65336 | 71509 | 393 |
| 628 | 890506 | 72130 | 72404 | 393 |
| 629 | 890506 | 73021 | 74150 | 303 |
| 630 | 890506 | 74807 | 75936 | 213 |
| 631 | 890506 | 80553 | 83119 | 123 |
| 632 | 890506 | 83736 | 84906 | 213 |
| 633 | 890506 | 85522 | 90652 | 303 |
| 634 | 890506 | 91308 | 93835 | 393 |

| | | | | |
|---|---|---|---|---|
| 635 | 890506 | 94103 | 95733 | 393 |
| 636 | 890506 | 140327 | 142459 | 393 |
| 637 | 890506 | 143121 | 143356 | 393 |
| 638 | 890506 | 144012 | 145141 | 303 |
| 639 | 890506 | 145758 | 150929 | 213 |
| 640 | 890506 | 151545 | 154111 | 123 |
| 641 | 890506 | 154728 | 155858 | 213 |
| 642 | 890506 | 160514 | 161644 | 303 |
| 643 | 890506 | 162300 | 164828 | 393 |
| 644 | 890506 | 165056 | 171622 | 393 |
| 645 | 890506 | 171821 | 174348 | 393 |
| 646 | 890506 | 174615 | 181141 | 393 |
| 647 | 890506 | 181758 | 182928 | 303 |
| 648 | 890507 | 55811 | 62338 | 393 |
| 649 | 890507 | 62604 | 65131 | 393 |
| 650 | 890507 | 65747 | 70916 | 303 |
| 651 | 890507 | 71532 | 72700 | 213 |
| 652 | 890507 | 73317 | 75844 | 123 |
| 653 | 890507 | 80460 | 81629 | 213 |
| 654 | 890507 | 82245 | 83414 | 303 |
| 655 | 890507 | 84031 | 90557 | 393 |
| 656 | 890507 | 90824 | 93350 | 393 |
| 657 | 890507 | 93549 | 100115 | 393 |
| 658 | 890507 | 100343 | 102908 | 393 |
| 659 | 890507 | 103525 | 104654 | 303 |
| 660 | 890507 | 105310 | 110439 | 213 |
| 661 | 890507 | 111056 | 113622 | 123 |
| 662 | 890507 | 114239 | 115408 | 213 |
| 663 | 890507 | 120024 | 121153 | 303 |
| 664 | 890507 | 121810 | 124336 | 393 |
| 665 | 890507 | 124604 | 131130 | 393 |
| 666 | 890507 | 131328 | 133855 | 393 |
| 667 | 890507 | 134122 | 140648 | 393 |
| 668 | 890507 | 141304 | 142434 | 303 |
| 669 | 890507 | 143050 | 144219 | 213 |
| 670 | 890507 | 144835 | 151402 | 123 |
| 671 | 890507 | 152018 | 153148 | 213 |
| 672 | 890507 | 153804 | 154932 | 303 |
| 673 | 890507 | 155549 | 162116 | 393 |
| 674 | 890507 | 162343 | 164909 | 393 |
| 675 | 890507 | 165107 | 171634 | 393 |
| 676 | 890507 | 171900 | 174427 | 393 |
| 677 | 890507 | 175043 | 180212 | 303 |
| 678 | 890507 | 180828 | 183455 | 213 |
| 679 | 890508 | 62106 | 64626 | 123 |
| 680 | 890508 | 65237 | 70404 | 213 |
| 681 | 890508 | 71015 | 72142 | 303 |
| 682 | 890508 | 72753 | 75313 | 393 |
| 683 | 890508 | 75538 | 82058 | 393 |
| 684 | 890508 | 82254 | 84813 | 393 |
| 685 | 890508 | 85038 | 91600 | 393 |
| 686 | 890508 | 92216 | 93346 | 303 |
| 687 | 890508 | 94002 | 95131 | 213 |
| 688 | 890508 | 95748 | 102314 | 123 |

| | | | | | |
|---|---|---|---|---|---|
| 689 | 890508 | 103326 | 104456 | 213 | |
| 690 | 890508 | 111120 | 113647 | 393 | |
| 691 | 890508 | 113915 | 120441 | 393 | |
| 692 | 890508 | 120639 | 123206 | 393 | |
| 693 | 890508 | 123433 | 125959 | 393 | |
| 694 | 890508 | 130616 | 131745 | 303 | |
| 695 | 890508 | 132401 | 133530 | 213 | |
| 696 | 890508 | 134147 | 140712 | 123 | |
| 697 | 890508 | 141329 | 142458 | 213 | |
| 698 | 890508 | 143115 | 144243 | 303 | |
| 699 | 890508 | 144859 | 151427 | 393 | |
| 700 | 890508 | 151654 | 154220 | 393 | |
| 701 | 890508 | 154419 | 160050 | 393 | |
| 702 | 890508 | 160712 | 160945 | 393 | |
| 703 | 890508 | 161213 | 163739 | 393 | |
| 704 | 890508 | 164354 | 165524 | 303 | |
| 705 | 890508 | 170140 | 171309 | 213 | |
| 706 | 890508 | 171926 | 174453 | 123 | |
| 707 | 890508 | 175108 | 180238 | 213 | |
| 708 | 890508 | 180853 | 182022 | 303 | |
| 709 | 890508 | 182638 | 185225 | 393 | |
| 710 | 890509 | 70847 | 73019 | 393 | |
| 711 | 890509 | 73641 | 73916 | 393 | |
| 712 | 890509 | 74531 | 75701 | 303 | |
| 713 | 890509 | 80317 | 80552 | 213 | |
| 714 | 890509 | 81213 | 81454 | 123 | ? |
| 715 | 890509 | 82124 | 82959 | 213 | ? |
| 716 | 890509 | 83221 | 84853 | 123 | ? |
| 717 | 890509 | 85510 | 90639 | 213 | |
| 718 | 890509 | 91254 | 92425 | 303 | |
| 719 | 890509 | 93040 | 95607 | 393 | |
| 720 | 890509 | 95835 | 102402 | 393 | |
| 721 | 890509 | 102601 | 105127 | 393 | |
| 722 | 890509 | 105355 | 111921 | 393 | |
| 723 | 890509 | 112714 | 113844 | 303 | |
| 724 | 890600 | 120000 | 120000 | 0 | END OF FILE FLAG |

# Appendix II

A listing of the scattering point parameter "tape" files SPP - GR XXX, which are the disc files corresponding to the original MAPSTAR data tapes numbered XXX.Also listed are the data date and hour, and number of scattering points between 66 and 116km recorded in the *previous* hour.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| SPP - GR | 45 | 89. | 3. | 28. | 1800 | 0 | SCATTERING POINTS |
| SPP - GR | 45 | 89. | 3. | 28. | 1900 | 124 | SCATTERING POINTS |
| SPP - GR | 45 | 89. | 3. | 28. | 2000 | 4522 | SCATTERING POINTS |
| SPP - GR | 46 | 89. | 3. | 28. | 2100 | 3201 | SCATTERING POINTS |
| SPP - GR | 47 | 89. | 3. | 28. | 2200 | 4185 | SCATTERING POINTS |
| SPP - GR | 47 | 89. | 3. | 28. | 2300 | 4383 | SCATTERING POINTS |
| SPP - GR | 48 | 89. | 3. | 29. | 0000 | 1836 | SCATTERING POINTS |
| SPP - GR | 49 | 89. | 3. | 29. | 0100 | 1030 | SCATTERING POINTS |
| SPP - GR | 49 | 89. | 3. | 29. | 0200 | 2497 | SCATTERING POINTS |
| SPP - GR | 50 | 89. | 3. | 29. | 0300 | 2291 | SCATTERING POINTS |
| SPP - GR | 51 | 89. | 3. | 29. | 0400 | 3525 | SCATTERING POINTS |
| SPP - GR | 51 | 89. | 3. | 29. | 0500 | 5690 | SCATTERING POINTS |
| SPP - GR | 52 | 89. | 3. | 29. | 0600 | 4459 | SCATTERING POINTS |
| SPP - GR | 53 | 89. | 3. | 29. | 0700 | 11776 | SCATTERING POINTS |
| SPP - GR | 53 | 89. | 3. | 29. | 0800 | 14499 | SCATTERING POINTS |
| SPP - GR | 54 | 89. | 3. | 29. | 0900 | 9260 | SCATTERING POINTS |
| SPP - GR | 55 | 89. | 3. | 29. | 1000 | 8909 | SCATTERING POINTS |
| SPP - GR | 55 | 89. | 3. | 29. | 1100 | 8733 | SCATTERING POINTS |
| SPP - GR | 56 | 89. | 3. | 29. | 1200 | 11050 | SCATTERING POINTS |
| SPP - GR | 56 | 89. | 3. | 29. | 1300 | 11733 | SCATTERING POINTS |
| SPP - GR | 57 | 89. | 3. | 29. | 1400 | 9709 | SCATTERING POINTS |
| SPP - GR | 58 | 89. | 3. | 29. | 1500 | 8529 | SCATTERING POINTS |
| SPP - GR | 59 | 89. | 3. | 29. | 1600 | 6965 | SCATTERING POINTS |
| SPP - GR | 60 | 89. | 3. | 29. | 1700 | 11103 | SCATTERING POINTS |
| SPP - GR | 60 | 89. | 3. | 29. | 1800 | 3606 | SCATTERING POINTS |
| SPP - GR | 61 | 89. | 3. | 29. | 1900 | 5820 | SCATTERING POINTS |
| SPP - GR | 61 | 89. | 3. | 29. | 2000 | 4436 | SCATTERING POINTS |
| SPP - GR | 62 | 89. | 3. | 29. | 2100 | 4468 | SCATTERING POINTS |
| SPP - GR | 63 | 89. | 3. | 29. | 2200 | 14625 | SCATTERING POINTS |
| SPP - GR | 63 | 89. | 3. | 29. | 2300 | 14147 | SCATTERING POINTS |
| SPP - GR | 64 | 89. | 3. | 30. | 0000 | 3438 | SCATTERING POINTS |
| SPP - GR | 65 | 89. | 3. | 30. | 0100 | 1185 | SCATTERING POINTS |
| SPP - GR | 66 | 89. | 3. | 30. | 0200 | 3201 | SCATTERING POINTS |
| SPP - GR | 66 | 89. | 3. | 30. | 0300 | 2336 | SCATTERING POINTS |
| SPP - GR | 67 | 89. | 3. | 30. | 0400 | 2511 | SCATTERING POINTS |
| SPP - GR | 68 | 89. | 3. | 30. | 0500 | 2859 | SCATTERING POINTS |
| SPP - GR | 68 | 89. | 3. | 30. | 0600 | 2708 | SCATTERING POINTS |
| SPP - GR | 69 | 89. | 3. | 30. | 0700 | 1611 | SCATTERING POINTS |
| SPP - GR | 69 | 89. | 3. | 30. | 0800 | 3030 | SCATTERING POINTS |
| SPP - GR | 69 | 89. | 3. | 30. | 0900 | 12626 | SCATTERING POINTS |
| SPP - GR | 70 | 89. | 3. | 30. | 1000 | 10461 | SCATTERING POINTS |
| SPP - GR | 71 | 89. | 3. | 30. | 1100 | 9236 | SCATTERING POINTS |
| SPP - GR | 71 | 89. | 3. | 30. | 1200 | 4956 | SCATTERING POINTS |
| SPP - GR | 71 | 89. | 3. | 30. | 1300 | 3225 | SCATTERING POINTS |
| SPP - GR | 72 | 89. | 3. | 30. | 1400 | 2039 | SCATTERING POINTS |
| SPP - GR | 72 | 89. | 3. | 30. | 1500 | 964 | SCATTERING POINTS |
| SPP - GR | 72 | 89. | 3. | 30. | 1600 | 3060 | SCATTERING POINTS |
| SPP - GR | 73 | 89. | 3. | 30. | 1700 | 6911 | SCATTERING POINTS |
| SPP - GR | 74 | 89. | 3. | 30. | 1800 | 9243 | SCATTERING POINTS |
| SPP - GR | 74 | 89. | 3. | 30. | 1900 | 6677 | SCATTERING POINTS |
| SPP - GR | 75 | 89. | 3. | 30. | 2000 | 1834 | SCATTERING POINTS |
| SPP - GR | 76 | 89. | 3. | 30. | 2100 | 1100 | SCATTERING POINTS |
| SPP - GR | 76 | 89. | 3. | 30. | 2200 | 5823 | SCATTERING POINTS |
| SPP - GR | 77 | 89. | 3. | 30. | 2300 | 5909 | SCATTERING POINTS |
| SPP - GR | 78 | 89. | 3. | 31. | 0000 | 11999 | SCATTERING POINTS |
| SPP - GR | 78 | 89. | 3. | 31. | 0100 | 7054 | SCATTERING POINTS |
| SPP - GR | 79 | 89. | 3. | 31. | 0200 | 4703 | SCATTERING POINTS |
| SPP - GR | 80 | 89. | 3. | 31. | 0300 | 1298 | SCATTERING POINTS |
| SPP - GR | 80 | 89. | 3. | 31. | 0400 | 1266 | SCATTERING POINTS |
| SPP - GR | 81 | 89. | 3. | 31. | 0500 | 5113 | SCATTERING POINTS |
| SPP - GR | 82 | 89. | 3. | 31. | 0600 | 4787 | SCATTERING POINTS |
| SPP - GR | 82 | 89. | 3. | 31. | 0700 | 12394 | SCATTERING POINTS |
| SPP - GR | 83 | 89. | 3. | 31. | 0800 | 13202 | SCATTERING POINTS |
| SPP - GR | 84 | 89. | 3. | 31. | 0900 | 13003 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 84 | 89. | 3. | 31. | 1000 | 13038 | SCATTERING POINTS |
| SPP - GR 85 | 89. | 3. | 31. | 1100 | 11332 | SCATTERING POINTS |
| SPP - GR 86 | 89. | 3. | 31. | 1200 | 11122 | SCATTERING POINTS |
| SPP - GR 86 | 89. | 3. | 31. | 1300 | 13417 | SCATTERING POINTS |
| SPP - GR 87 | 89. | 3. | 31. | 1400 | 11780 | SCATTERING POINTS |
| SPP - GR 88 | 89. | 3. | 31. | 1500 | 11306 | SCATTERING POINTS |
| SPP - GR 89 | 89. | 3. | 31. | 1600 | 10354 | SCATTERING POINTS |
| SPP - GR 89 | 89. | 3. | 31. | 1700 | 10123 | SCATTERING POINTS |
| SPP - GR 90 | 89. | 3. | 31. | 1800 | 11582 | SCATTERING POINTS |
| SPP - GR 91 | 89. | 3. | 31. | 1900 | 10709 | SCATTERING POINTS |
| SPP - GR 91 | 89. | 3. | 31. | 2000 | 8235 | SCATTERING POINTS |
| SPP - GR 92 | 89. | 3. | 31. | 2100 | 6441 | SCATTERING POINTS |
| SPP - GR 93 | 89. | 3. | 31. | 2200 | 7091 | SCATTERING POINTS |
| SPP - GR 93 | 89. | 3. | 31. | 2300 | 10953 | SCATTERING POINTS |
| SPP - GR 94 | 89. | 4. | 1. | 0000 | 9177 | SCATTERING POINTS |
| SPP - GR 95 | 89. | 4. | 1. | 0100 | 6691 | SCATTERING POINTS |
| SPP - GR 95 | 89. | 4. | 1. | 0200 | 8231 | SCATTERING POINTS |
| SPP - GR 96 | 89. | 4. | 1. | 0300 | 6754 | SCATTERING POINTS |
| SPP - GR 97 | 89. | 4. | 1. | 0400 | 10161 | SCATTERING POINTS |
| SPP - GR 97 | 89. | 4. | 1. | 0500 | 7319 | SCATTERING POINTS |
| SPP - GR 98 | 89. | 4. | 1. | 0600 | 5931 | SCATTERING POINTS |
| SPP - GR 99 | 89. | 4. | 1. | 0600 | 5218 | SCATTERING POINTS |
| SPP - GR 99 | 89. | 4. | 1. | 0700 | 4877 | SCATTERING POINTS |
| SPP - GR 99 | 89. | 4. | 1. | 0800 | 22972 | SCATTERING POINTS |
| SPP - GR 100 | 89. | 4. | 1. | 0900 | 21179 | SCATTERING POINTS |
| SPP - GR 101 | 89. | 4. | 1. | 1000 | 22363 | SCATTERING POINTS |
| SPP - GR 101 | 89. | 4. | 1. | 1100 | 24097 | SCATTERING POINTS |
| SPP - GR 102 | 89. | 4. | 1. | 1200 | 3205 | SCATTERING POINTS |
| SPP - GR 102 | 89. | 4. | 1. | 1300 | 21481 | SCATTERING POINTS |
| SPP - GR 103 | 89. | 4. | 1. | 1400 | 24053 | SCATTERING POINTS |
| SPP - GR 103 | 89. | 4. | 1. | 1500 | 3473 | SCATTERING POINTS |
| SPP - GR 103 | 89. | 4. | 1. | 1600 | 28734 | SCATTERING POINTS |
| SPP - GR 104 | 89. | 4. | 1. | 1700 | 18901 | SCATTERING POINTS |
| SPP - GR 105 | 89. | 4. | 1. | 1800 | 15120 | SCATTERING POINTS |
| SPP - GR 105 | 89. | 4. | 1. | 1900 | 8908 | SCATTERING POINTS |
| SPP - GR 106 | 89. | 4. | 1. | 2000 | 2948 | SCATTERING POINTS |
| SPP - GR 107 | 89. | 4. | 1. | 2100 | 8139 | SCATTERING POINTS |
| SPP - GR 107 | 89. | 4. | 1. | 2200 | 15322 | SCATTERING POINTS |
| SPP - GR 108 | 89. | 4. | 1. | 2300 | 4407 | SCATTERING POINTS |
| SPP - GR 109 | 89. | 4. | 2. | 0000 | 6711 | SCATTERING POINTS |
| SPP - GR 109 | 89. | 4. | 2. | 0100 | 11299 | SCATTERING POINTS |
| SPP - GR 110 | 89. | 4. | 2. | 0200 | 5654 | SCATTERING POINTS |
| SPP - GR 111 | 89. | 4. | 2. | 0300 | 7220 | SCATTERING POINTS |
| SPP - GR 111 | 89. | 4. | 2. | 0400 | 6366 | SCATTERING POINTS |
| SPP - GR 112 | 89. | 4. | 2. | 0500 | 10171 | SCATTERING POINTS |
| SPP - GR 113 | 89. | 4. | 2. | 0600 | 8629 | SCATTERING POINTS |
| SPP - GR 113 | 89. | 4. | 2. | 0700 | 18642 | SCATTERING POINTS |
| SPP - GR 114 | 89. | 4. | 2. | 0800 | 4263 | SCATTERING POINTS |
| SPP - GR 114 | 89. | 4. | 2. | 0900 | 12119 | SCATTERING POINTS |
| SPP - GR 115 | 89. | 4. | 2. | 1000 | 22881 | SCATTERING POINTS |
| SPP - GR 115 | 89. | 4. | 2. | 1100 | 24797 | SCATTERING POINTS |
| SPP - GR 116 | 89. | 4. | 2. | 1200 | 18282 | SCATTERING POINTS |
| SPP - GR 117 | 89. | 4. | 2. | 1300 | 20341 | SCATTERING POINTS |
| SPP - GR 118 | 89. | 4. | 2. | 1500 | 16177 | SCATTERING POINTS |
| SPP - GR 118 | 89. | 4. | 2. | 1600 | 4174 | SCATTERING POINTS |
| SPP - GR 118 | 89. | 4. | 2. | 1700 | 20598 | SCATTERING POINTS |
| SPP - GR 119 | 89. | 4. | 2. | 1800 | 13256 | SCATTERING POINTS |
| SPP - GR 120 | 89. | 4. | 2. | 1900 | 11120 | SCATTERING POINTS |
| SPP - GR 120 | 89. | 4. | 2. | 2000 | 8341 | SCATTERING POINTS |
| SPP - GR 121 | 89. | 4. | 2. | 2100 | 7506 | SCATTERING POINTS |
| SPP - GR 122 | 89. | 4. | 2. | 2200 | 4661 | SCATTERING POINTS |
| SPP - GR 122 | 89. | 4. | 2. | 2300 | 6587 | SCATTERING POINTS |
| SPP - GR 123 | 89. | 4. | 3. | 0000 | 6098 | SCATTERING POINTS |
| SPP - GR 124 | 89. | 4. | 3. | 0100 | 3580 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 124 | 89. | 4. | 3. | 0200 | 3049 | SCATTERING POINTS |
| SPP - GR 125 | 89. | 4. | 3. | 0300 | 8230 | SCATTERING POINTS |
| SPP - GR 125 | 89. | 4. | 3. | 0400 | 11324 | SCATTERING POINTS |
| SPP - GR 126 | 89. | 4. | 3. | 0500 | 11824 | SCATTERING POINTS |
| SPP - GR 127 | 89. | 4. | 3. | 0600 | 12506 | SCATTERING POINTS |
| SPP - GR 127 | 89. | 4. | 3. | 0700 | 18238 | SCATTERING POINTS |
| SPP - GR 128 | 89. | 4. | 3. | 0800 | 19989 | SCATTERING POINTS |
| SPP - GR 129 | 89. | 4. | 3. | 1000 | 18375 | SCATTERING POINTS |
| SPP - GR 129 | 89. | 4. | 3. | 1100 | 15741 | SCATTERING POINTS |
| SPP - GR 130 | 89. | 4. | 3. | 1200 | 18570 | SCATTERING POINTS |
| SPP - GR 131 | 89. | 4. | 3. | 1300 | 18546 | SCATTERING POINTS |
| SPP - GR 131 | 89. | 4. | 3. | 1400 | 20168 | SCATTERING POINTS |
| SPP - GR 132 | 89. | 4. | 3. | 1500 | 18879 | SCATTERING POINTS |
| SPP - GR 133 | 89. | 4. | 3. | 1600 | 16111 | SCATTERING POINTS |
| SPP - GR 133 | 89. | 4. | 3. | 1700 | 23191 | SCATTERING POINTS |
| SPP - GR 134 | 89. | 4. | 3. | 1800 | 18935 | SCATTERING POINTS |
| SPP - GR 135 | 89. | 4. | 3. | 1900 | 13342 | SCATTERING POINTS |
| SPP - GR 135 | 89. | 4. | 3. | 2000 | 5740 | SCATTERING POINTS |
| SPP - GR 136 | 89. | 4. | 3. | 2100 | 2440 | SCATTERING POINTS |
| SPP - GR 137 | 89. | 4. | 3. | 2200 | 10917 | SCATTERING POINTS |
| SPP - GR 137 | 89. | 4. | 3. | 2300 | 9477 | SCATTERING POINTS |
| SPP - GR 138 | 89. | 4. | 4. | 0000 | 10078 | SCATTERING POINTS |
| SPP - GR 139 | 89. | 4. | 4. | 0100 | 6287 | SCATTERING POINTS |
| SPP - GR 139 | 89. | 4. | 4. | 0200 | 12047 | SCATTERING POINTS |
| SPP - GR 140 | 89. | 4. | 4. | 0300 | 6277 | SCATTERING POINTS |
| SPP - GR 141 | 89. | 4. | 4. | 0400 | 6133 | SCATTERING POINTS |
| SPP - GR 142 | 89. | 4. | 4. | 0500 | 6042 | SCATTERING POINTS |
| SPP - GR 142 | 89. | 4. | 4. | 0600 | 14207 | SCATTERING POINTS |
| SPP - GR 143 | 89. | 4. | 4. | 0700 | 21747 | SCATTERING POINTS |
| SPP - GR 144 | 89. | 4. | 4. | 0900 | 26671 | SCATTERING POINTS |
| SPP - GR 144 | 89. | 4. | 4. | 1000 | 3868 | SCATTERING POINTS |
| SPP - GR 144 | 89. | 4. | 4. | 1100 | 23428 | SCATTERING POINTS |
| SPP - GR 145 | 89. | 4. | 4. | 1200 | 18363 | SCATTERING POINTS |
| SPP - GR 146 | 89. | 4. | 4. | 1300 | 21877 | SCATTERING POINTS |
| SPP - GR 146 | 89. | 4. | 4. | 1400 | 23588 | SCATTERING POINTS |
| SPP - GR 147 | 89. | 4. | 4. | 1500 | 18609 | SCATTERING POINTS |
| SPP - GR 148 | 89. | 4. | 4. | 1600 | 12921 | SCATTERING POINTS |
| SPP - GR 149 | 89. | 4. | 4. | 1700 | 4017 | SCATTERING POINTS |
| SPP - GR 149 | 89. | 4. | 4. | 1800 | 16914 | SCATTERING POINTS |
| SPP - GR 150 | 89. | 4. | 4. | 1900 | 10176 | SCATTERING POINTS |
| SPP - GR 151 | 89. | 4. | 4. | 2000 | 7170 | SCATTERING POINTS |
| SPP - GR 151 | 89. | 4. | 4. | 2100 | 10889 | SCATTERING POINTS |
| SPP - GR 152 | 89. | 4. | 4. | 2200 | 7139 | SCATTERING POINTS |
| SPP - GR 153 | 89. | 4. | 4. | 2300 | 10088 | SCATTERING POINTS |
| SPP - GR 153 | 89. | 4. | 5. | 0000 | 11742 | SCATTERING POINTS |
| SPP - GR 154 | 89. | 4. | 5. | 0100 | 10109 | SCATTERING POINTS |
| SPP - GR 155 | 89. | 4. | 5. | 0200 | 9409 | SCATTERING POINTS |
| SPP - GR 155 | 89. | 4. | 5. | 0300 | 5727 | SCATTERING POINTS |
| SPP - GR 156 | 89. | 4. | 5. | 0400 | 7750 | SCATTERING POINTS |
| SPP - GR 157 | 89. | 4. | 5. | 0500 | 12021 | SCATTERING POINTS |
| SPP - GR 157 | 89. | 4. | 5. | 0600 | 12790 | SCATTERING POINTS |
| SPP - GR 158 | 89. | 4. | 5. | 0700 | 23139 | SCATTERING POINTS |
| SPP - GR 159 | 89. | 4. | 5. | 0800 | 20998 | SCATTERING POINTS |
| SPP - GR 160 | 89. | 4. | 5. | 0900 | 25640 | SCATTERING POINTS |
| SPP - GR 160 | 89. | 4. | 5. | 1000 | 16473 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 161 | 89. | 4. | 5. | 1100 | 26068 | SCATTERING POINTS |
| SPP - GR 161 | 89. | 4. | 5. | 1200 | 33275 | SCATTERING POINTS |
| SPP - GR 162 | 89. | 4. | 5. | 1300 | 26963 | SCATTERING POINTS |
| SPP - GR 163 | 89. | 4. | 5. | 1400 | 23106 | SCATTERING POINTS |
| SPP - GR 163 | 89. | 4. | 5. | 1500 | 21578 | SCATTERING POINTS |
| SPP - GR 164 | 89. | 4. | 5 | 1600 | 19271 | SCATTERING POINTS |
| SPP - GR 165 | 89. | 4. | 5. | 1700 | 17394 | SCATTERING POINTS |
| SPP - GR 165 | 89. | 4. | 5. | 1800 | 17392 | SCATTERING POINTS |
| SPP - GR 166 | 89. | 4. | 5. | 1900 | 7803 | SCATTERING POINTS |
| SPP - GR 167 | 89. | 4. | 5. | 2000 | 2720 | SCATTERING POINTS |
| SPP - GR 167 | 89. | 4. | 5. | 2100 | 11035 | SCATTERING POINTS |
| SPP - GR 168 | 89. | 4. | 5. | 2200 | 15844 | SCATTERING POINTS |
| SPP - GR 169 | 89. | 4. | 5. | 2300 | 8885 | SCATTERING POINTS |
| SPP - GR 170 | 89. | 4. | 6. | 0000 | 14891 | SCATTERING POINTS |
| SPP - GR 170 | 89. | 4. | 6. | 0100 | 12061 | SCATTERING POINTS |
| SPP - GR 171 | 89. | 4. | 6. | 0200 | 13343 | SCATTERING POINTS |
| SPP - GR 172 | 89. | 4. | 6. | 0300 | 14350 | SCATTERING POINTS |
| SPP - GR 172 | 89. | 4. | 6. | 0400 | 10705 | SCATTERING POINTS |
| SPP - GR 173 | 89. | 4. | 6. | 0500 | 9101 | SCATTERING POINTS |
| SPP - GR 174 | 89. | 4. | 6. | 0700 | 6696 | SCATTERING POINTS |
| SPP - GR 174 | 89. | 4. | 6. | 0800 | 5583 | SCATTERING POINTS |
| SPP - GR 174 | 89. | 4. | 6. | 0900 | 19943 | SCATTERING POINTS |
| SPP - GR 175 | 89. | 4. | 6. | 1000 | 22871 | SCATTERING POINTS |
| SPP - GR 176 | 89. | 4. | 6. | 1100 | 23320 | SCATTERING POINTS |
| SPP - GR 176 | 89. | 4. | 6. | 1200 | 23139 | SCATTERING POINTS |
| SPP - GR 177 | 89. | 4. | 6. | 1300 | 19919 | SCATTERING POINTS |
| SPP - GR 178 | 89. | 4. | 6. | 1400 | 18610 | SCATTERING POINTS |
| SPP - GR 179 | 89. | 4. | 6. | 1500 | 21856 | SCATTERING POINTS |
| SPP - GR 179 | 89. | 4. | 6. | 1600 | 14893 | SCATTERING POINTS |
| SPP - GR 180 | 89. | 4. | 6. | 1700 | 16017 | SCATTERING POINTS |
| SPP - GR 181 | 89. | 4. | 6. | 1800 | 13495 | SCATTERING POINTS |
| SPP - GR 181 | 89. | 4. | 6. | 1900 | 8159 | SCATTERING POINTS |
| SPP - GR 182 | 89. | 4. | 6. | 2000 | 4663 | SCATTERING POINTS |
| SPP - GR 183 | 89. | 4. | 6. | 2100 | 4617 | SCATTERING POINTS |
| SPP - GR 183 | 89. | 4. | 6. | 2200 | 5325 | SCATTERING POINTS |
| SPP - GR 184 | 89. | 4. | 6. | 2300 | 6070 | SCATTERING POINTS |
| SPP - GR 185 | 89. | 4. | 7. | 0000 | 5978 | SCATTERING POINTS |
| SPP - GR 185 | 89. | 4. | 7. | 0100 | 17197 | SCATTERING POINTS |
| SPP - GR 186 | 89. | 4. | 7. | 0200 | 9273 | SCATTERING POINTS |
| SPP - GR 187 | 89. | 4. | 7. | 0300 | 10027 | SCATTERING POINTS |
| SPP - GR 187 | 89. | 4. | 7. | 0400 | 11959 | SCATTERING POINTS |
| SPP - GR 188 | 89. | 4. | 7. | 0500 | 8771 | SCATTERING POINTS |
| SPP - GR 189 | 89. | 4. | 7. | 0600 | 3055 | SCATTERING POINTS |
| SPP - GR 189 | 89. | 4. | 7. | 0700 | 1443 | SCATTERING POINTS |
| SPP - GR 189 | 89. | 4. | 7. | 0800 | 8433 | SCATTERING POINTS |
| SPP - GR 190 | 89. | 4. | 7. | 0900 | 15621 | SCATTERING POINTS |
| SPP - GR 191 | 89. | 4. | 7. | 1000 | 13454 | SCATTERING POINTS |
| SPP - GR 191 | 89. | 4. | 7. | 1100 | 4974 | SCATTERING POINTS |
| SPP - GR 192 | 89. | 4. | 7. | 1200 | 9879 | SCATTERING POINTS |
| SPP - GR 193 | 89. | 4. | 7. | 1300 | 24610 | SCATTERING POINTS |
| SPP - GR 193 | 89. | 4. | 7. | 1400 | 20688 | SCATTERING POINTS |
| SPP - GR 194 | 89. | 4. | 7. | 1500 | 13801 | SCATTERING POINTS |
| SPP - GR 195 | 89. | 4. | 7. | 1600 | 8986 | SCATTERING POINTS |
| SPP - GR 195 | 89. | 4. | 7. | 1700 | 7830 | SCATTERING POINTS |
| SPP - GR 196 | 89. | 4. | 7. | 1800 | 7290 | SCATTERING POINTS |
| SPP - GR 197 | 89. | 4. | 7. | 1900 | 5377 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 197 | 89 | 4 | 7. | 2000 | 8469 | SCATTERING POINTS |
| SPP - GR 198 | 89 | 4. | 7. | 2100 | 10535 | SCATTERING POINTS |
| SPP - GR 199 | 89 | 4. | 7. | 2200 | 10495 | SCATTERING POINTS |
| SPP - GR 199 | 89 | 4. | 7. | 2300 | 15950 | SCATTERING POINTS |
| SPP - GR 200 | 89 | 4. | 8. | 0000 | 16833 | SCATTERING POINTS |
| SPP - GR 201 | 89. | 4. | 8 | 0100 | 12430 | SCATTERING POINTS |
| SPP - GR 201 | 89 | 4. | 8. | 0200 | 18070 | SCATTERING POINTS |
| SPP - GR 202 | 89. | 4. | 8. | 0300 | 12185 | SCATTERING POINTS |
| SPP - GR 203 | 89 | 4. | 8. | 0400 | 16297 | SCATTERING POINTS |
| SPP - GR 203 | 89 | 4 | 8. | 0500 | 19409 | SCATTERING POINTS |
| SPP - GR 204 | 89. | 4. | 8. | 0600 | 12945 | SCATTERING POINTS |
| SPP - GR 205 | 89. | 4 | 8. | 0700 | 13820 | SCATTERING POINTS |
| SPP - GR 205 | 89. | 4. | 8. | 0800 | 17333 | SCATTERING POINTS |
| SPP - GR 206 | 89. | 4. | 8. | 0900 | 15909 | SCATTERING POINTS |
| SPP - GR 207 | 89. | 4. | 8. | 1000 | 19966 | SCATTERING POINTS |
| SPP - GR 207 | 89. | 4. | 8. | 1100 | 25304 | SCATTERING POINTS |
| SPP - GR 208 | 89. | 4. | 8. | 1200 | 20020 | SCATTERING POINTS |
| SPP - GR 209 | 89. | 4. | 8. | 1300 | 19227 | SCATTERING POINTS |
| SPP - GR 209 | 89. | 4. | 8. | 1400 | 20352 | SCATTERING POINTS |
| SPP - GR 210 | 89. | 4. | 8. | 1500 | 15218 | SCATTERING POINTS |
| SPP - GR 211 | 89. | 4. | 8. | 1600 | 13476 | SCATTERING POINTS |
| SPP - GR 211 | 89. | 4. | 8. | 1700 | 13007 | SCATTERING POINTS |
| SPP - GR 212 | 89. | 4. | 8. | 1800 | 14255 | SCATTERING POINTS |
| SPP - GR 213 | 89. | 4. | 8. | 1900 | 9923 | SCATTERING POINTS |
| SPP - GR 214 | 89. | 4. | 8. | 2000 | 10210 | SCATTERING POINTS |
| SPP - GR 214 | 89. | 4. | 8. | 2100 | 6596 | SCATTERING POINTS |
| SPP - GR 215 | 89. | 4. | 8. | 2200 | 11267 | SCATTERING POINTS |
| SPP - GR 216 | 89. | 4. | 8. | 2300 | 9731 | SCATTERING POINTS |
| SPP - GR 216 | 89. | 4. | 9. | 0000 | 5447 | SCATTERING POINTS |
| SPP - GR 217 | 89. | 4. | 9. | 0100 | 6756 | SCATTERING POINTS |
| SPP - GR 218 | 89. | 4. | 9. | 0200 | 6316 | SCATTERING POINTS |
| SPP - GR 218 | 89. | 4. | 9. | 0300 | 9872 | SCATTERING POINTS |
| SPP - GR 219 | 89. | 4. | 9. | 0400 | 11270 | SCATTERING POINTS |
| SPP - GR 220 | 89. | 4. | 9. | 0500 | 12399 | SCATTERING POINTS |
| SPP - GR 220 | 89. | 4. | 9. | 0600 | 9564 | SCATTERING POINTS |
| SPP - GR 221 | 89. | 4. | 9. | 0700 | 18037 | SCATTERING POINTS |
| SPP - GR 221 | 89. | 4. | 9. | 0800 | 23490 | SCATTERING POINTS |
| SPP - GR 222 | 89. | 4. | 9. | 0900 | 21060 | SCATTERING POINTS |
| SPP - GR 223 | 89. | 4. | 9. | 1000 | 14868 | SCATTERING POINTS |
| SPP - GR 223 | 89. | 4. | 9. | 1100 | 17839 | SCATTERING POINTS |
| SPP - GR 224 | 89. | 4. | 9. | 1200 | 7539 | SCATTERING POINTS |
| SPP - GR 225 | 89. | 4. | 9. | 1300 | 17460 | SCATTERING POINTS |
| SPP - GR 225 | 89. | 4. | 9. | 1400 | 16862 | SCATTERING POINTS |
| SPP - GR 226 | 89. | 4. | 9. | 1500 | 13275 | SCATTERING POINTS |
| SPP - GR 227 | 89. | 4. | 9. | 1600 | 12418 | SCATTERING POINTS |
| SPP - GR 227 | 89. | 4. | 9. | 1700 | 9096 | SCATTERING POINTS |
| SPP - GR 228 | 89. | 4. | 9. | 1800 | 7455 | SCATTERING POINTS |
| SPP - GR 229 | 89. | 4. | 9. | 1900 | 13737 | SCATTERING POINTS |
| SPP - GR 229 | 89. | 4. | 9. | 2000 | 4760 | SCATTERING POINTS |
| SPP - GR 230 | 89. | 4. | 9. | 2100 | 5142 | SCATTERING POINTS |
| SPP - GR 231 | 89. | 4. | 9. | 2200 | 13490 | SCATTERING POINTS |
| SPP - GR 231 | 89. | 4. | 9. | 2300 | 16239 | SCATTERING POINTS |
| SPP - GR 232 | 89. | 4. | 10. | 0000 | 11484 | SCATTERING POINTS |
| SPP - GR 233 | 89. | 4. | 10. | 0100 | 9396 | SCATTERING POINTS |
| SPP - GR 233 | 89. | 4. | 10. | 0200 | 8593 | SCATTERING POINTS |
| SPP - GR 234 | 89. | 4. | 10. | 0300 | 15477 | SCATTERING POINTS |
| SPP - GR 235 | 89. | 4. | 10. | 0400 | 8490 | SCATTERING POINTS |
| SPP - GR 235 | 89. | 4. | 10. | 0500 | 9614 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 236 | 89. | 4. | 10 | 0600 | 12087 | SCATTERING POINTS |
| SPP - GR 237 | 89. | 4 | 10 | 0700 | 14089 | SCATTERING POINTS |
| SPP - GR 237 | 89 | 4. | 10. | 0800 | 11768 | SCATTERING POINTS |
| SPP - GR 238 | 89. | 4. | 10. | 0900 | 5687 | SCATTERING POINTS |
| SPP - GR 238 | 89. | 4. | 10 | 1000 | 17624 | SCATTERING POINTS |
| SPP - GR 239 | 89 | 4. | 10. | 1100 | 9710 | SCATTERING POINTS |
| SPP - GR 239 | 89. | 4 | 10 | 1200 | 17650 | SCATTERING POINTS |
| SPP - GR 240 | 89. | 4. | 10. | 1300 | 13225 | SCATTERING POINTS |
| SPP - GR 241 | 89 | 4 | 10 | 1400 | 16109 | SCATTERING POINTS |
| SPP - GR 241 | 89. | 4. | 10. | 1500 | 15634 | SCATTERING POINTS |
| SPP - GR 242 | 89. | 4. | 10. | 1600 | 15928 | SCATTERING POINTS |
| SPP - GR 243 | 89. | 4. | 10. | 1700 | 13047 | SCATTERING POINTS |
| SPP - GR 243 | 89. | 4. | 10. | 1800 | 12643 | SCATTERING POINTS |
| SPP - GR 245 | 89. | 4. | 10. | 1900 | 9097 | SCATTERING POINTS |
| SPP - GR 245 | 89. | 4. | 10. | 2000 | 3250 | SCATTERING POINTS |
| SPP - GR 245 | 89. | 4. | 10. | 2100 | 6495 | SCATTERING POINTS |
| SPP - GR 246 | 89. | 4. | 10. | 2200 | 5529 | SCATTERING POINTS |
| SPP - GR 247 | 89. | 4. | 10. | 2300 | 17396 | SCATTERING POINTS |
| SPP - GR 248 | 89. | 4. | 11. | 0000 | 11674 | SCATTERING POINTS |
| SPP - GR 248 | 89. | 4. | 11. | 0100 | 10545 | SCATTERING POINTS |
| SPP - GR 249 | 89. | 4. | 11. | 0200 | 6706 | SCATTERING POINTS |
| SPP - GR 249 | 89. | 4. | 11. | 0300 | 17942 | SCATTERING POINTS |
| SPP - GR 250 | 89. | 4. | 11. | 0400 | 19293 | SCATTERING POINTS |
| SPP - GR 251 | 89. | 4. | 11. | 0500 | 10177 | SCATTERING POINTS |
| SPP - GR 251 | 89. | 4. | 11. | 0600 | 18170 | SCATTERING POINTS |
| SPP - GR 252 | 89. | 4. | 11. | 0700 | 21042 | SCATTERING POINTS |
| SPP - GR 253 | 89. | 4. | 11. | 0800 | 21212 | SCATTERING POINTS |
| SPP - GR 253 | 89. | 4. | 11. | 0900 | 19311 | SCATTERING POINTS |
| SPP - GR 254 | 89. | 4. | 11. | 1000 | 18809 | SCATTERING POINTS |
| SPP - GR 255 | 89. | 4. | 11. | 1100 | 18967 | SCATTERING POINTS |
| SPP - GR 256 | 89. | 4. | 11. | 1600 | 10319 | SCATTERING POINTS |
| SPP - GR 257 | 89. | 5. | 2. | 1700 | 2142 | SCATTERING POINTS |
| SPP - GR 257 | 89. | 5. | 2. | 1800 | 1424 | SCATTERING POINTS |
| SPP - GR 258 | 89. | 5. | 2. | 1900 | 1546 | SCATTERING POINTS |
| SPP GR 258 | 89. | 5. | 2. | 2000 | 1980 | SCATTERING POINTS |
| SPP - GR 259 | 89. | 5. | 2. | 2100 | 2590 | SCATTERING POINTS |
| SPP - GR 259 | 89. | 5. | 2. | 2200 | 1403 | SCATTERING POINTS |
| SPP - GR 260 | 89. | 5. | 2. | 2300 | 1829 | SCATTERING POINTS |
| SPP - GR 260 | 89. | 5. | 3. | 0000 | 2158 | SCATTERING POINTS |
| SPP - GR 261 | 89. | 5. | 3. | 0100 | 4175 | SCATTERING POINTS |
| SPP - GR 261 | 89. | 5. | 3. | 0200 | 2775 | SCATTERING POINTS |
| SPP - GR 262 | 89. | 5. | 3. | 0300 | 2865 | SCATTERING POINTS |
| SPP - GR 263 | 89. | 5. | 3. | 0400 | 5987 | SCATTERING POINTS |
| SPP - GR 263 | 89. | 5. | 3. | 0500 | 4179 | SCATTERING POINTS |
| SPP - GR 263 | 89. | 5. | 3. | 0600 | 6843 | SCATTERING POINTS |
| SPP - GR 264 | 89. | 5. | 3. | 0700 | 4975 | SCATTERING POINTS |
| SPP - GR 264 | 89. | 5. | 3. | 0800 | 7225 | SCATTERING POINTS |
| SPP - GR 265 | 89. | 5. | 3. | 0900 | 6657 | SCATTERING POINTS |
| SPP - GR 265 | 89. | 5. | 3. | 1000 | 6131 | SCATTERING POINTS |
| SPP - GR 266 | 89. | 5. | 3. | 1100 | 6735 | SCATTERING POINTS |
| SPP - GR 267 | 89. | 5. | 3. | 1200 | 4715 | SCATTERING POINTS |
| SPP - GR 267 | 89. | 5. | 3. | 1300 | 5174 | SCATTERING POINTS |
| SPP - GR 268 | 89. | 5. | 3. | 1400 | 4832 | SCATTERING POINTS |
| SPP - GR 268 | 89. | 5. | 3. | 1500 | 4111 | SCATTERING POINTS |
| SPP - GR 269 | 89. | 5. | 3. | 1600 | 7444 | SCATTERING POINTS |
| SPP - GR 269 | 89. | 5. | 3. | 1700 | 7617 | SCATTERING POINTS |
| SPP - GR 270 | 89. | 5. | 3. | 1800 | 6223 | SCATTERING POINTS |
| SPP - GR 270 | 89. | 5. | 3. | 1900 | 5529 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 271 | 89. | 5. | 3. | 2000 | 6513 | SCATTERING POINTS |
| SPP - GR 271 | 89. | 5. | 3. | 2100 | 5458 | SCATTERING POINTS |
| SPP - GR 272 | 89. | 5. | 3. | 2200 | 7629 | SCATTERING POINTS |
| SPP - GR 272 | 89. | 5. | 3. | 2300 | 7595 | SCATTERING POINTS |
| SPP - GR 273 | 89. | 5. | 4. | 0000 | 7374 | SCATTERING POINTS |
| SPP - GR 273 | 89. | 5. | 4. | 0100 | 4191 | SCATTERING POINTS |
| SPP - GR 274 | 89. | 5. | 4. | 0200 | 3260 | SCATTERING POINTS |
| SPP - GR 274 | 89. | 5. | 4. | 0300 | 3744 | SCATTERING POINTS |
| SPP - GR 275 | 89. | 5. | 4. | 0400 | 4298 | SCATTERING POINTS |
| SPP - GR 275 | 89. | 5. | 4. | 0500 | 3750 | SCATTERING POINTS |
| SPP - GR 276 | 89. | 5. | 4. | 0600 | 4326 | SCATTERING POINTS |
| SPP - GR 276 | 89. | 5. | 4. | 0700 | 5264 | SCATTERING POINTS |
| SPP - GR 277 | 89. | 5. | 4. | 0800 | 4778 | SCATTERING POINTS |
| SPP - GR 277 | 89. | 5. | 4. | 0900 | 5826 | SCATTERING POINTS |
| SPP - GR 278 | 89. | 5. | 4. | 1000 | 3535 | SCATTERING POINTS |
| SPP - GR 278 | 89. | 5. | 4. | 1100 | 3868 | SCATTERING POINTS |
| SPP - GR 279 | 89. | 5. | 4. | 1200 | 4382 | SCATTERING POINTS |
| SPP - GR 279 | 89. | 5. | 4. | 1300 | 721 | SCATTERING POINTS |
| SPP - GR 280 | 89. | 5. | 4. | 1400 | 1693 | SCATTERING POINTS |
| SPP - GR 280 | 89. | 5. | 4. | 1500 | 796 | SCATTERING POINTS |
| SPP - GR 281 | 89. | 5. | 4. | 1600 | 1019 | SCATTERING POINTS |
| SPP - GR 281 | 89. | 5. | 4. | 1700 | 2361 | SCATTERING POINTS |
| SPP - GR 282 | 89. | 5. | 4. | 1800 | 3310 | SCATTERING POINTS |
| SPP - GR 282 | 89 | 5. | 4. | 1900 | 3157 | SCATTERING POINTS |
| SPP - GR 283 | 89. | 5. | 4. | 2000 | 7320 | SCATTERING POINTS |
| SPP - GR 284 | 89. | 5. | 4. | 2100 | 11013 | SCATTERING POINTS |
| SPP - GR 284 | 89. | 5. | 4. | 2200 | 5430 | SCATTERING POINTS |
| SPP - GR 285 | 89. | 5. | 4. | 2300 | 5145 | SCATTERING POINTS |
| SPP - GR 285 | 89. | 5. | 5. | 0000 | 4218 | SCATTERING POINTS |
| SPP - GR 286 | 89. | 5. | 5. | 0100 | 2511 | SCATTERING POINTS |
| SPP - GR 286 | 89. | 5. | 5. | 0200 | 2698 | SCATTERING POINTS |
| SPP - GR 287 | 89. | 5. | 5. | 0300 | 4075 | SCATTERING POINTS |
| SPP - GR 287 | 89. | 5. | 5. | 0400 | 2803 | SCATTERING POINTS |
| SPP - GR 288 | 89. | 5. | 5. | 0500 | 4000 | SCATTERING POINTS |
| SPP - GR 288 | 89. | 5. | 5. | 0600 | 2484 | SCATTERING POINTS |
| SPP - GR 289 | 89. | 5. | 5. | 0700 | 2755 | SCATTERING POINTS |
| SPP - GR 289 | 89. | 5. | 5. | 0800 | 5911 | SCATTERING POINTS |
| SPP - GR 290 | 89. | 5. | 5. | 0900 | 5870 | SCATTERING POINTS |
| SPP - GR 290 | 89. | 5. | 5. | 1000 | 6867 | SCATTERING POINTS |
| SPP - GR 291 | 89. | 5. | 5. | 1100 | 3528 | SCATTERING POINTS |
| SPP - GR 292 | 89. | 5. | 5. | 1200 | 1467 | SCATTERING POINTS |
| SPP - GR 292 | 89. | 5. | 5. | 1300 | 7820 | SCATTERING POINTS |
| SPP - GR 293 | 89. | 5. | 5. | 1400 | 3229 | SCATTERING POINTS |
| SPP - GR 293 | 89. | 5. | 5. | 1500 | 3388 | SCATTERING POINTS |
| SPP - GR 294 | 89. | 5. | 5. | 1600 | 4872 | SCATTERING POINTS |
| SPP - GR 294 | 89. | 5. | 5. | 1700 | 3182 | SCATTERING POINTS |
| SPP - GR 295 | 89. | 5. | 5. | 1800 | 6839 | SCATTERING POINTS |
| SPP - GR 295 | 89. | 5. | 5. | 1900 | 7009 | SCATTERING POINTS |
| SPP - GR 296 | 89. | 5. | 5. | 2000 | 2387 | SCATTERING POINTS |
| SPP - GR 296 | 89. | 5. | 5. | 2100 | 2408 | SCATTERING POINTS |
| SPP - GR 297 | 89. | 5. | 5. | 2200 | 4236 | SCATTERING POINTS |
| SPP - GR 297 | 89. | 5. | 5. | 2300 | 5798 | SCATTERING POINTS |
| SPP - GR 298 | 89. | 5. | 6. | 0000 | 7668 | SCATTERING POINTS |
| SPP - GR 298 | 89. | 5. | 6. | 0100 | 5067 | SCATTERING POINTS |
| SPP - GR 299 | 89. | 5. | 6. | 0200 | 3296 | SCATTERING POINTS |
| SPP - GR 299 | 89. | 5. | 6. | 0300 | 4678 | SCATTERING POINTS |
| SPP - GR 300 | 89. | 5. | 6. | 0400 | 11144 | SCATTERING POINTS |
| SPP - GR 300 | 89. | 5. | 6. | 0500 | 6825 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 301 | 89 | 5 | 6. | 0600 | 18295 | SCATTERING POINTS |
| SPP - GR 301 | 89 | 5 | 6 | 0700 | 7881 | SCATTERING POINTS |
| SPP - GR 302 | 89. | 5. | 6. | 0600 | 10105 | SCATTERING POINTS |
| SPP - GR 302 | 89 | 5 | 6 | 0900 | 10130 | SCATTERING POINTS |
| SPP - GR 303 | 89 | 5 | 6 | 1000 | 1427 | SCATTERING POINTS |
| SPP - GR 303 | 89 | 5 | 6 | 1100 | 917 | SCATTERING POINTS |
| SPP - GR 304 | 89 | 5 | 6 | 1200 | 4464 | SCATTERING POINTS |
| SPP - GR 304 | 89 | 5 | 6 | 1300 | 4165 | SCATTERING POINTS |
| SPP - GR 305 | 89 | 5 | 6 | 1400 | 5519 | SCATTERING POINTS |
| SPP - GR 305 | 89 | 5 | 6 | 1500 | 6018 | SCATTERING POINTS |
| SPP - GR 306 | 89. | 5 | 6. | 1600 | 6028 | SCATTERING POINTS |
| SPP - GR 306 | 89. | 5 | 6. | 1700 | 6395 | SCATTERING POINTS |
| SPP - GR 307 | 89. | 5. | 6. | 1800 | 5639 | SCATTERING POINTS |
| SPP - GR 307 | 89. | 5. | 6. | 1900 | 9355 | SCATTERING POINTS |
| SPP - GR 308 | 89. | 5. | 6. | 2000 | 7113 | SCATTERING POINTS |
| SPP - GR 308 | 89. | 5. | 6. | 2100 | 3388 | SCATTERING POINTS |
| SPP - GR 309 | 89. | 5. | 6. | 2200 | 6611 | SCATTERING POINTS |
| SPP - GR 309 | 89. | 5. | 6. | 2300 | 5374 | SCATTERING POINTS |
| SPP - GR 310 | 89. | 5. | 7 | 0000 | 2932 | SCATTERING POINTS |
| SPP - GR 310 | 89. | 5. | 7. | 0100 | 4267 | SCATTERING POINTS |
| SPP - GR 311 | 89. | 5. | 7. | 0200 | 4537 | SCATTERING POINTS |
| SPP - GR 311 | 89. | 5. | 7. | 0300 | 4099 | SCATTERING POINTS |
| SPP - GR 312 | 89. | 5. | 7. | 0400 | 4326 | SCATTERING POINTS |
| SPP - GR 312 | 89. | 5. | 7. | 0500 | 3232 | SCATTERING POINTS |
| SPP - GR 313 | 89. | 5. | 7. | 0600 | 4964 | SCATTERING POINTS |
| SPP - GR 313 | 89. | 5. | 7. | 0700 | 5256 | SCATTERING POINTS |
| SPP - GR 314 | 89. | 5. | 7. | 0800 | 6073 | SCATTERING POINTS |
| SPP - GR 314 | 89. | 5. | 7. | 0900 | 11023 | SCATTERING POINTS |
| SPP - GR 315 | 89. | 5. | 7. | 1000 | 9978 | SCATTERING POINTS |
| SPP - GR 315 | 89. | 5. | 7. | 1100 | 8169 | SCATTERING POINTS |
| SPP - GR 316 | 89. | 5. | 7. | 1200 | 8321 | SCATTERING POINTS |
| SPP - GR 316 | 89. | 5. | 7. | 1300 | 11472 | SCATTERING POINTS |
| SPP - GR 317 | 89. | 5. | 7. | 1400 | 8636 | SCATTERING POINTS |
| SPP - GR 317 | 89. | 5. | 7. | 1500 | 7391 | SCATTERING POINTS |
| SPP - GR 318 | 89. | 5. | 7. | 1600 | 6701 | SCATTERING POINTS |
| SPP - GR 318 | 89. | 5. | 7 | 1700 | 6309 | SCATTERING POINTS |
| SPP - GR 319 | 89. | 5. | 7. | 1800 | 4184 | SCATTERING POINTS |
| SPP - GR 319 | 89. | 5. | 7. | 1900 | 5511 | SCATTERING POINTS |
| SPP - GR 320 | 89. | 5. | 7. | 2000 | 1588 | SCATTERING POINTS |
| SPP - GR 320 | 89. | 5. | 7. | 2100 | 5026 | SCATTERING POINTS |
| SPP - GR 321 | 89. | 5. | 7. | 2200 | 3639 | SCATTERING POINTS |
| SPP - GR 321 | 89 | 5. | 7. | 2300 | 3655 | SCATTERING POINTS |
| SPP - GR 322 | 89. | 5. | 8. | 0000 | 10926 | SCATTERING POINTS |
| SPP - GR 322 | 89. | 5. | 8. | 0100 | 5579 | SCATTERING POINTS |
| SPP - GR 323 | 89. | 5. | 8. | 0200 | 2079 | SCATTERING POINTS |
| SPP - GR 323 | 89. | 5. | 8. | 0300 | 3846 | SCATTERING POINTS |
| SPP - GR 324 | 89. | 5. | 8. | 0400 | 4082 | SCATTERING POINTS |
| SPP - GR 324 | 89. | 5. | 8. | 0500 | 3610 | SCATTERING POINTS |
| SPP - GR 325 | 89. | 5. | 8. | 0600 | 4551 | SCATTERING POINTS |
| SPP - GR 325 | 89. | 5. | 8. | 0700 | 5470 | SCATTERING POINTS |
| SPP - GR 326 | 89. | 5. | 8. | 0800 | 7715 | SCATTERING POINTS |
| SPP - GR 326 | 89. | 5. | 8. | 0900 | 12069 | SCATTERING POINTS |
| SPP - GR 327 | 89. | 5. | 8. | 1000 | 11089 | SCATTERING POINTS |
| SPP - GR 328 | 89. | 5. | 8. | 1200 | 10768 | SCATTERING POINTS |
| SPP - GR 328 | 89. | 5. | 8. | 1300 | 7909 | SCATTERING POINTS |
| SPP - GR 329 | 89. | 5. | 8. | 1400 | 5931 | SCATTERING POINTS |
| SPP - GR 329 | 89. | 5. | 8. | 1500 | 7599 | SCATTERING POINTS |
| SPP - GR 330 | 89. | 5. | 8. | 1600 | 5436 | SCATTERING POINTS |

| | | | | | | |
|---|---|---|---|---|---|---|
| SPP - GR 330 | 89 | 5 | 8 | 1700 | 15183 | SCATTERING POINTS |
| SPP - GR 331 | 89. | 5. | 8 | 1800 | 6390 | SCATTERING POINTS |
| SPP - GR 332 | 89. | 5. | 8 | 1900 | 8455 | SCATTERING POINTS |
| SPP - GR 332 | 89 | 5. | 8. | 2000 | 13600 | SCATTERING POINTS |
| SPP - GR 333 | 89. | 5. | 8 | 2100 | 10171 | SCATTERING POINTS |
| SPP - GR 333 | 89 | 5 | 8 | 2200 | 12110 | SCATTERING POINTS |
| SPP - GR 334 | 89 | 5. | 8 | 2300 | 7115 | SCATTERING POINTS |
| SPP - GR 335 | 89 | 5. | 9. | 0000 | 17068 | SCATTERING POINTS |
| SPP - GR 335 | 89. | 5. | 9. | 0100 | 8490 | SCATTERING POINTS |
| SPP - GR 336 | 89. | 5 | 9. | 0200 | 5021 | SCATTERING POINTS |
| SPP - GR 336 | 89. | 5. | 9. | 0300 | 6695 | SCATTERING POINTS |
| SPP - GR 337 | 89. | 5. | 9. | 0400 | 4885 | SCATTERING POINTS |
| SPP - GR 337 | 89. | 5. | 9. | 0500 | 5825 | SCATTERING POINTS |
| SPP - GR 338 | 89. | 5. | 9. | 0600 | 4973 | SCATTERING POINTS |
| SPP - GR 339 | 89. | 5. | 9. | 0700 | 5111 | SCATTERING POINTS |
| SPP - GR 339 | 89. | 5. | 9. | 0800 | 10634 | SCATTERING POINTS |
| SPP - GR 340 | 89. | 5. | 9. | 0900 | 10231 | SCATTERING POINTS |
| SPP - GR 340 | 89. | 5. | 9. | 1000 | 17392 | SCATTERING POINTS |
| SPP - GR 341 | 89. | 5. | 9. | 1100 | 10772 | SCATTERING POINTS |